A

MAJOR PROJECT REPORT ON

IMPLEMENTATION OF WEARABLE DEVICE FOR CHILD SAFETY

Submitted in partial fulfilment of the requirement for the award of degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

SUBMITTED BY

PULLURI SRI RAM GOUD	**218R1A04B2**
RATHOD KRISHNA	**218R1A04B3**
REVELLY RAJU	**218R1A04B4**
SUDHA SWATHI	**218R1A04B5**

Under the Esteemed Guidance of

Mr. P. CHANDER
Assistant professor

DEPARTMENT OF ELECTRONICS&COMMUNICATION ENGINEERING

CMR ENGINEERING COLLEGE

UGC AUTONOMOUS

(Approved by AICTE, Affiliated to JNTU Hyderabad, Accredited by NBA)

Kandlakoya(V), Medchal(M), Telangana – 501401

(2024-2025)

CMR ENGINEERING COLLEGE

UGC AUTONOMOUS

(Approved by AICTE, Affiliated to JNTU Hyderabad, Accredited by NBA)

Kandlakoya(V), Medchal Road, Hyderabad - 501 401

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that the major-project work entitled **"IMPLEMENTATION OF WEARABLE DEVICES FOR CHILD SAFETY"** is being submitted by **P. SRIRAM** bearing Roll No **218R1A04B2, R. KRISHNA** bearing Roll No **218R1A04B3, R. RAJU** bearing Roll No **218R1A04B4, S. SWATHI** bearing Roll No **218R1A04B5** in B.Tech IV-II semester, Electronics and Communication Engineering is a record Bonafide work carried out during the academic year 2024-25. The results embodied in this report have not been submitted to any other University for the award of any degree.

INTERNAL GUIDE	HEAD OF THE DEPARTMENT
Mr. P. CHANDER	**Dr. SUMAN MISHRA**

EXTERNAL EXAMINER

ACKNOWLEDGEMENTS

We sincerely thank the management of our college **CMR Engineering College** for providing required facilities during our project work. We derive great pleasure in expressing our sincere gratitude to our Principal **Dr. A. S. REDDY** for his timely suggestions, which helped us to complete the project work successfully. It is the very auspicious moment we would like to express our gratitude to **Dr. SUMAN MISHRA,** Head of the Department, ECE for his consistent encouragement during the progress of this project.

We take it as a privilege to thank our project coordinator **Dr. T. SATYANARAYANA**, Associate Professor, Department of ECE for the ideas that led to complete the project work and we also thank him for his continuous guidance, support and unfailing patience, throughout the course of this work. We sincerely thank our project internal guide **Mr. P. CHANDER,** Assistant Professor of ECE for guidance and encouragement in carrying out this project work.

DECLARATION

We hereby declare that the major project entitled **“IMPLEMENTATION OF WEARABLE DEVICES FOR CHILD SAFETY”** is the work done by us in campus at **CMR ENGINEERING COLLEGE,** Kandlakoya during the academic year 2024-2025 and is submitted as major project in partial fulfilment of the requirements for the award of degree of **BACHELOR OF TECHNOLOGY** in **ELECTRONICS AND COMMUNICATION ENGINEERING FROM JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, HYDERABAD.**

P. SRI RAM	**(218R1A04B2)**
R. KRISHNA	**(218R1A04B3)**
R. RAJU	**(218R1A04B4)**
S. SWATHI	**(218R1A04B5)**

ABSTRACT

This project presents a prototype of a wearable child safety device that integrates various sensors and communication technologies to ensure the safety and well-being of children. The device utilizes a panic button, GSM module, MEMS (Micro-Electro-Mechanical Systems) sensors, temperature sensor, heartbeat sensor, IoT module, and LED indicators to provide real-time monitoring and immediate alerts. The panic button, when pressed by the child, sends an SOS signal to the parent’s mobile phone, notifying them of an emergency situation. The GSM module transmits the child's location data, enabling parents to track their child's movements in real time, while the MEMS sensor detects any sudden movements or falls, triggering an alert.

In addition to safety monitoring, the wearable device also tracks the child’s health using the temperature and heartbeat sensors. The temperature sensor continuously monitors the child's body temperature, and the heartbeat sensor tracks heart rate, sending notifications if abnormal readings are detected. The IoT module ensures seamless communication between the wearable device and the parent’s smartphone, providing constant access to the child’s status. LED indicators on the device offer a visual cue to both the child and parents, signaling the device’s operational status or an emergency.

This child safety wearable prototype integrates multiple features to ensure a comprehensive monitoring system that addresses both location safety and health. The use of IoT connectivity and real-time alerts makes the device a powerful tool for preventing and responding to emergencies. Overall, the prototype provides an efficient and innovative solution to enhance child safety by combining location tracking, health monitoring, and immediate communication in one wearable device.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER-1
INTRODUCTION

This project presents a wearable child protection device that incorporates cutting-edge technologies for better security. The device has a panic button, GSM module, MEMS sensors to detect falls, temperature and heartbeat sensors to monitor health, an IoT module for real-time connectivity, and LED indicators for status alerts visually. It allows parents to monitor the location and health of their child in real time and send immediate alerts in situations of emergency. With the incorporation of all these features, the device guarantees safety as well as wellness, enabling parents to respond in a timely manner to any signals of distress or illness, offering an exhaustive solution for child safeguarding.

1.1 EMBEDDED SYSTEM

Embedded systems typically consist of both hardware and software components. The hardware includes microcontrollers or microprocessors, sensors, actuators, memory, and communication interfaces, which enable the system to interact with external devices and the environment. The software, often referred to as firmware, is stored in non-volatile memory and controls the behavior of the embedded system. It is responsible for processing sensor data, managing hardware components, and communicating with other systems when necessary. Many embedded systems are designed to work without a full operating system, though more complex applications may use real-time operating systems (RTOS) to manage multiple tasks simultaneously.

A significant feature of embedded systems is their ability to perform tasks in real time, meaning they must process inputs and respond to them within strict time limits. For example, in a child safety wearable device, the embedded system is responsible for continuously monitoring the child’s location, health parameters, and activity, providing immediate responses to emergencies, and sending alerts to parents. The system must process data from various sensors, such as heart rate monitors, temperature sensors, and MEMS sensors for fall detection, ensuring the child’s safety in real-time.

In the case of a child safety wearable, embedded systems integrate several technologies, including a microcontroller to process data, sensors for health and motion monitoring, and communication modules like GSM or IoT for sending alerts. The panic button allows the child to send an SOS signal, while the system tracks location and health parameters. The embedded system in the wearable ensures low power consumption, reliability, and efficiency, enabling the device to function for extended periods. Additionally, LED indicators on the device provide visual status cues, allowing both parents and children to understand the device’s operational state.

Overall, embedded systems are integral to the functioning of modern devices, providing the necessary infrastructure for real-time processing, communication, and monitoring in applications like child safety wearables. By combining hardware and software in a compact, efficient manner, embedded systems ensure that such devices perform their dedicated tasks effectively and reliably.

1.2 HISTORY AND FUTURE

Wearable devices for child safety have come a long way since their inception, and their future promises to bring even more advanced features and functionalities. In the past, the safety of children was primarily ensured through physical means like ID tags, wristbands, or verbal instructions, which were often inadequate in emergency situations. The introduction of technology in the early 2000s brought GPS-based trackers that allowed parents to monitor their children's location remotely. These devices were often bulky and limited in features, but they laid the groundwork for the integration of more advanced technologies into wearable safety devices.

As technology evolved, the 2010s witnessed a rapid expansion in the capabilities of wearable devices. The combination of GPS, Bluetooth, and mobile app integration revolutionized child safety by providing real-time location tracking, geofencing, and emergency alerts directly to parents’ smartphones. Devices like the Jiobit and TickTalk, for example, offered a variety of safety features, including real-time location tracking, communication via text or voice calls, and emergency buttons that allowed children to instantly notify their parents if they were in trouble. These wearables became smaller, more durable, and user-friendly, catering to the specific needs of children.

In the present day, wearables for children are equipped with even more advanced features. Some of these devices include sensors that monitor a child's health by tracking heart rate, steps, and even detecting falls or impacts. If a child has a sudden fall or exhibits signs of distress, the device automatically sends an alert to parents or emergency contacts, ensuring a quick response. The devices now also integrate advanced geofencing technology that allows parents to create safe zones around particular locations, such as their home or school, and receive notifications when their child enters or exits these areas.

Looking to the future, the potential for wearable devices for child safety is vast. The incorporation of artificial intelligence (AI) and machine learning (ML) could significantly enhance the predictive capabilities of these devices. AI could analyze patterns in a child's behavior, such as rapid changes in movement, abnormal heart rate, or even location patterns, to predict and prevent dangerous situations. These devices could send real-time alerts to parents or caregivers if they detect unusual activity or if the child is approaching a hazardous area. In addition, the use of biometric sensors may allow these devices to monitor a child's overall health and safety in ways that go beyond what is possible today. For instance, wearables could monitor environmental factors like air quality or temperature, alerting parents if their child is in a potentially unsafe environment.

Moreover, future wearables are likely to become even more integrated with other smart technologies. They could communicate seamlessly with home security systems, smartphones, or even vehicles, enabling parents to ensure their child's safety in multiple contexts. For instance, the device could automatically notify parents when a child is approaching a car, or it could integrate with smart home systems to alert parents if a child is trying to access certain areas of the house. These devices could also provide better communication options, such as real-time video calling or even communication with emergency responders if needed.

With privacy and security being crucial concerns, future wearables for child safety will likely feature enhanced data encryption and tamper-proof designs to protect the information from hackers or misuse. Additionally, these devices will likely be designed to be more comfortable, durable, and child-friendly, ensuring that they are both practical and appealing to young users. As wearable technology continues to evolve, these devices will play an increasingly important role in keeping children safe, offering peace of mind to parents while fostering independence and confidence in their children.

1.3 REAL TIME SYSTEMS

Real-time systems (RTS) are specialized computing systems designed to respond to inputs within a precise timeframe. The key distinguishing feature of real-time systems is that they must guarantee timely responses, meaning they are structured to meet strict deadlines for processing data and executing tasks. In these systems, not meeting the specified deadlines can result in catastrophic consequences, particularly in safety-critical environments.

Real-time systems are categorized into hard**,** soft**,** and firm systems. Hard real-time systems require strict adherence to deadlines, where missing a deadline can cause complete failure or significant harm. Examples include flight control systems, pacemakers, and airbag deployment systems in cars, where every millisecond counts. Soft real-time systems**,** on the other hand, can tolerate occasional missed deadlines without serious consequences. Video streaming, live gaming, or multimedia applications fall under this category, where a slight delay may degrade quality but doesn't cause the system to fail. Firm real-time systems represent a middle ground, where missing deadlines doesn’t cause total failure, but it does affect performance, like in real-time financial trading systems.

The architecture of real-time systems requires a careful balance between several factors. Task scheduling is fundamental, where real-time operating systems (RTOS) use algorithms like Rate-Monotonic Scheduling (RMS), Earliest Deadline First (EDF), or Least Laxity First (LLF) to determine which tasks should be executed first based on their urgency and deadlines. Real-time systems must also handle concurrency, as many tasks often run in parallel, requiring careful synchronization to avoid conflicts. Synchronization mechanisms, such as semaphores or mutexes, are employed to ensure tasks don't interfere with one another.

In real-time systems, interrupt handling plays a crucial role in responding to critical events quickly. An interrupt is a signal from hardware or software that demands the immediate attention of the system, which then temporarily suspends ongoing tasks to address the high-priority event. This feature is essential in applications such as medical devices or automotive safety, where a real-time response is required to avoid disasters. Resource management is another crucial aspect, ensuring that limited resources like memory and processing power are allocated effectively so that deadlines are met.

One of the biggest challenges for real-time systems is timing and synchronization**,** especially in multi-core or distributed systems. For example, in distributed systems, tasks might be spread across different nodes, and delays in communication can affect timing. Reliability and fault tolerance are also key concerns, as real-time systems often operate in mission-critical environments where failures can have serious consequences. To mitigate these risks, many real-time systems employ redundancy, failover mechanisms, and self-checking processes to ensure the system can recover from partial failures.

The applications of real-time systems span across numerous industries. In automotive systems**,** real-time systems are essential for safety features like collision detection and automated braking, where delays could result in accidents. Medical devices such as pacemakers, infusion pumps, and heart rate monitors depend on real-time systems to provide continuous, accurate data and perform life-saving actions. In telecommunication**s**, real-time systems handle voice calls, video conferencing, and data transmission, ensuring minimal delay and optimal performance. In industrial control, real-time systems manage machinery and production lines, ensuring that automated tasks are carried out with high precision and without interruption. Aerospace and defense applications rely on real-time systems for flight control, missile guidance, and satellite communication, where performance must be flawless to prevent catastrophic failures.

As the world moves towards more interconnected, data-driven environments, the future of real-time systems looks promising. Edge computing and 5G networks are expected to significantly reduce latency by bringing computational resources closer to data sources, enabling real-time processing at the edge of the network. This will enhance real-time decision-making in applications like autonomous vehicles, smart cities, and industrial automation. Moreover, with the rise of Artificial Intelligence (AI)**,** real-time systems may evolve to become more adaptive and predictive. AI could help anticipate and prevent issues before they occur, by analyzing patterns and adjusting system behavior in real-time. Additionally, the increasing adoption of Internet of Things (IoT) devices will create more data points for real-time systems to monitor and act upon, making them even more integral to modern technological ecosystems.

However, the growing complexity of these systems brings with it new challenges. As real-time systems become more interconnected and distributed, ensuring security and managing data integrity becomes a critical concern. Malicious attacks, such as denial-of-service attacks, could potentially disrupt time-sensitive operations, making the need for robust cybersecurity measures even more urgent.

In conclusion, real-time systems are integral to a wide range of applications that require timely, precise responses to external inputs. They are a cornerstone of modern critical systems, such as healthcare, transportation, telecommunications, and defense, and their importance will only continue to grow as technologies like AI, edge computing, and IoT continue to advance. The evolution of these systems will shape the future of numerous industries, ensuring that we can meet increasingly demanding technological and societal needs in real-time.

1.4 OVERVIEW OF PROJECT

This project focuses on developing a wearable child safety device that uses real-time systems to ensure the continuous monitoring and protection of children in various environments, such as at school, in public places, or even at home. The device is designed to provide both physical and digital safeguards by integrating multiple technologies, including GPS tracking, health monitoring, emergency response features, and communication systems. The overall goal is to provide parents and guardians with real-time, actionable information to keep their children safe.

Core Features of the Wearable Device

1. **GPS Tracking and Geofencing**:
    - The device will feature real-time GPS tracking, which enables the exact location of the child to be monitored by parents through a mobile app. This ensures that parents always know where their child is, even in crowded environments or unfamiliar places.
    - The geofencing technology allows parents to set safe zones (e.g., home, school, or park). If the child exits these boundaries, the system will automatically send an alert to the parent’s mobile device. This feature ensures that children don't wander off or enter potentially unsafe areas unnoticed.
2. **Health Monitoring**:
    - The wearable device will include sensors to monitor vital health metrics such as heart rate, body temperature, and step count. This feature will help ensure the child’s health is within safe parameters. In case of any unusual readings—such as a high or low heart rate—the system will notify the parent immediately, enabling quick intervention if necessary.
    - **Fall detection**: The device will also incorporate accelerometers to detect sudden falls or impacts. If the child falls or is in distress, the device can trigger an automatic alert to the guardian’s phone, offering a crucial layer of safety.
3. **Emergency Alert Button**:
    - In case of an emergency, the wearable device will have a panic button that allows the child to send an immediate alert to the parent or emergency contacts. When pressed, this button will activate a notification system to provide the child’s real-

time location along with the alert, ensuring that the guardian can act immediately.

4. **Mobile App Integration**:
    - The wearable device will communicate with a mobile app, enabling parents to monitor their child’s location, health data, and receive real-time alerts. This app will offer a user-friendly interface with features such as:
        - Live location tracking on a map
        - Viewing historical routes and activities of the child
        - Notifications for health alerts, such as abnormal vitals or fall detection
        - The ability to configure geofencing zones and receive exit alerts
        - Control over emergency settings and access to quick action buttons
    - The mobile app will serve as a central hub for monitoring the child’s safety and health status at all times.
5. **Battery Life and Durability**:
    - Since the wearable device will be used on children, it is essential that it has a long-lasting battery and is durable to withstand daily use, such as playing, swimming, or accidental impacts. The device will be designed to be waterproof and shock-resistant, ensuring that it is not only functional but also resilient.

Real-Time Systems Implementation

The core of the project lies in the real-time systems aspect, where the device’s responsiveness and ability to meet strict time constraints are vital. Several key elements will make this possible:

1. **Real-Time Processing**:
    - The device will utilize a real-time operating system (RTOS) that ensures quick processing of data and prioritizes critical tasks, such as health alerts or location updates, over less urgent ones. For example, if a fall is detected, the RTOS will immediately trigger an emergency alert, bypassing less important tasks like sending routine location updates.
2. **Task Scheduling and Prioritization**:
    - The system will use real-time scheduling algorithms, such as Rate-Monotonic Scheduling (RMS) or Earliest Deadline First (EDF), to prioritize tasks. High-priority events like location-based alerts, health warnings, or emergency buttons will be processed first, ensuring that there is no delay in critical situations.

3. **Data Synchronization**:
    - The wearable device will periodically synchronize with the mobile app to provide parents with real-time updates on their child's health, location, and overall status. This synchronization will occur in the background without interrupting other functions of the device, ensuring continuous monitoring with minimal user input.

Potential Future Enhancements

While the initial version of the wearable device will focus on location tracking, health monitoring, and emergency response, future iterations could expand into more advanced functionalities, such as:

1. **AI Integration**:
    - The use of artificial intelligence (AI) could allow the device to predict potential safety risks based on behavioral patterns. For instance, the system could learn when a child is most likely to wander off or exhibit unusual behavior, and preemptively alert the parent. AI could also help improve health monitoring by detecting early signs of distress or sickness.
2. **Voice Communication**:
    - Integrating voice communication into the device could allow the child and the parent to communicate directly via the wearable. This could be particularly useful if the child is unable to use a mobile phone but needs to alert the parent or ask for assistance.
3. **Social Media Integration**:
    - In the future, the device might integrate with social media or local community platforms, allowing the sharing of real-time location or distress alerts within trusted networks (e.g., friends, family, or emergency contacts).
4. **Advanced Health Features**:
    - Future models could incorporate more advanced biometric sensors, such as heart rate variability (HRV) monitors, oxygen level sensors, or electrocardiogram (ECG) capabilities. This would allow for more detailed health monitoring, particularly in detecting early signs of medical conditions such as asthma, arrhythmias, or stress.

CHAPTER-2

LITERATURE SURVEY

Wearable devices for child safety have become a significant area of research and development, leveraging advancements in real-time systems, GPS tracking, and health monitoring technologies. These devices are designed to ensure the safety and well-being of children by providing parents and guardians with real-time information regarding their child's location, health, and potential safety risks. A primary focus of such devices is real-time location tracking through GPS technology.

Devices like Jiobit and AngelSense utilize GPS, Wi-Fi, and Bluetooth to track a child’s location, allowing parents to monitor their child in public places or when they are not directly in sight. Research has demonstrated that GPS-based tracking provides peace of mind to parents, helping alleviate concerns about children wandering off or getting lost, particularly in crowded or unfamiliar environments.

In addition to location tracking**,** geofencing technology is employed to create virtual boundaries around safe zones, such as a school or home. If the child crosses these boundaries, an automatic alert is sent to the parent’s mobile device, ensuring immediate awareness of any potential danger. Studies have shown that geofencing can significantly enhance child safety by offering real-time updates and notifications when a child strays from predefined zones, allowing for timely interventions.

Another crucial aspect of child safety wearables is health monitoring. Many wearables integrate sensors to monitor a child’s vital signs, such as heart rate, body temperature, and even steps taken. This technology is particularly important for children with medical conditions or those engaged in physical activities, as it allows for continuous health tracking and early detection of abnormalities.

Devices like the Empatica Embrace2 focus on monitoring physiological data to alert parents of potential health emergencies. Research suggests that continuous biometric monitoring can prevent dangerous situations, such as undiagnosed health conditions or reactions to allergens, by sending immediate alerts when the child's health indicators deviate from normal ranges. Moreover, the integration of fall detection using accelerometers is a vital feature in many wearable devices. Sudden falls or accidents, especially in young children, can be critical, and wearable devices with fall detection can immediately notify guardians if an incident occurs. This real-time response can be crucial in preventing or mitigating injury, as the wearable device processes data rapidly to trigger an emergency alert.

The effectiveness of these devices relies heavily on the real-time systems that power them. Real-time operating systems (RTOS) ensure that time-sensitive tasks, such as location updates, health monitoring, and emergency alerts, are processed promptly and without delay. RTOS allows for efficient task scheduling, prioritizing urgent events like safety alerts over less critical activities.

Research highlights that using algorithms like Earliest Deadline First (EDF) or Rate-Monotonic Scheduling (RMS) ensures that high-priority tasks are executed without missing deadlines, which is crucial for child safety. Furthermore, communication protocols, such as Bluetooth Low Energy (BLE) and cellular networks, enable seamless data transfer between the wearable device and the parent’s mobile app, providing live updates and alerts. This communication ensures that parents receive accurate and timely information regarding their child’s safety status.

In conclusion, wearable devices for child safety, underpinned by real-time systems, GPS, geofencing, and health monitoring technologies, have shown great promise in enhancing child protection. These devices offer a multifaceted approach to safety, allowing parents to monitor their child's location, health, and well-being in real time. The integration of advanced technologies such as machine learning and AI could further enhance the predictive capabilities of these devices, providing even more proactive and intelligent safety measures. As these technologies continue to evolve, wearable devices for child safety will become even more reliable, offering parents greater peace of mind and ensuring children’s well-being in various environments.

2.1 EXISTING SYSTEM

Existing wearable systems for child safety that incorporate location history and two-way communication combine advanced tracking technologies and communication features to enhance child protection. Devices like Jiobit**,** AngelSense**,** TickTalk**,** Filip 2**,** and Garmin Bounce are designed to provide real-time tracking and communication between parents and children**.** Jiobit offers real-time GPS tracking along with location history, allowing parents to review where their child has been throughout the day. The device supports geofencing, which sends alerts when the child enters or leaves predefined areas, providing added security.

Jiobit also enables two-way communication, letting parents send and receive voice messages, ensuring they can stay in contact with their child anytime.

Angel Sense is designed primarily for children with special needs and includes features like location tracking, geofencing, and a listening-in function. Parents can access real-time location data, while the listening-in feature allows them to hear their child's surroundings and ensure they are safe. Additionally, the device supports two-way communication, enabling parents to talk to their child and vice versa.

Tick Talk, another smartwatch designed for children, integrates location tracking and geofencing, providing real-time updates on the child's location and sending alerts if they leave designated zones. This device also supports two-way communication through voice calls and video chats, allowing parents to stay connected at all times.

Filip 2 and Garmin Bounce offer real-time GPS tracking and location history, along with two-way communication through voice calls and text messaging. These devices also feature geofencing, notifying parents when their child crosses predefined boundaries. These systems provide a comprehensive approach to child safety, combining location tracking, geofencing, and two-way communication to ensure parents are always informed about their child's safety and can communicate in real-time in case of emergencies.

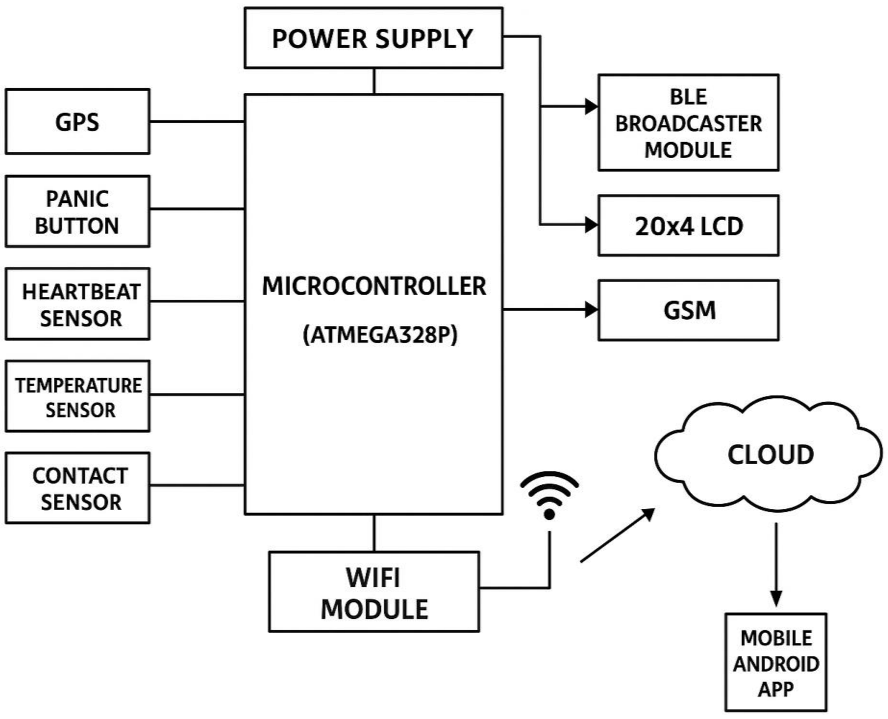


Fig 2.1: Existing Method

2.2 PROPOSED SYSTEMS

The proposed system is an advanced wearable child safety device designed to enhance child protection with AI-powered location tracking**,** dynamic geofencing, and improved two-way communication**.** Unlike existing systems, this device uses AI algorithms to analyze the child's movement patterns and predict any deviations from their normal route. If the child strays into unfamiliar or potentially unsafe areas, parents will receive instant alerts**.**

The dynamic geofencing feature adjusts the safe zones based on the time of day and the child's schedule, providing tailored alerts when the child crosses designated boundaries. In addition, the system will offer two-way communication, enabling voice calls**,** text messaging**,** and push-to-talk functionality for emergencies, ensuring quick and easy communication between parents and children.

Furthermore, the device will integrate health monitoring features, including heart rate monitoring and fall detection, sending alerts if any abnormal health events occur. To address battery life concerns, the device will utilize low power consumption technologies to ensure prolonged use without compromising tracking and communication capabilities. Finally, all data and communications will be secured with end-to-end encryption, ensuring privacy and safety for both the child and the parent. This system offers a more reliable, secure, and efficient solution for child safety.

2.3 EMBEDDED INTRODUCTION

Embedded systems are specialized computing systems designed to perform specific tasks within a larger system. Unlike general-purpose computers, which can run a wide range of applications, embedded systems are dedicated to specific functions and are often found in everyday devices such as smartphones, home appliances, medical equipment, and vehicles. These systems typically combine hardware and software to achieve a particular goal efficiently. An embedded system consists of a microcontroller or microprocessor at its core, which is responsible for processing data and controlling other components. These systems are designed to be real-time**,** meaning they must respond to inputs within a specific time frame, which is crucial for applications such as automated systems, medical devices, and safety-critical systems.

One of the main advantages of embedded systems is their reliability and efficiency. They are optimized to perform their tasks with minimal resource consumption, which makes them ideal for devices with limited processing power and memory. Additionally, embedded systems are often integrated into products, making them compact, cost-effective, and capable of providing long-term performance without the need for user intervention.

In modern technology, embedded systems are becoming increasingly complex, incorporating sensors, actuators, and communication modules to connect to other systems, enabling smart devices and contributing to the growth of the Internet of Things (IoT).

2.3.1 Specification

The project aims to develop a wearable child safety device using GPS for real-time location tracking and GSM for two-way communication, including alerts, geofencing, and emergency features to ensure child safety.

2.3.2 Prototyping

Prototyping tools and platforms like Arduino, Raspberry Pi, and ESP32 are commonly used to quickly build and test embedded systems before production. These platforms offer versatility, enabling rapid development and iteration of embedded applications.

2.3.3 Applications

Embedded systems are an integral part of modern technology, providing specialized functionality in a wide range of industries and devices. Some key applications of embedded systems include:

Consumer Electronics:

Embedded systems are widely used in consumer electronics, such as smartphones, smart TVs, digital cameras, smart home devices, and fitness trackers. These systems control features like display, sensors, connectivity, and power management, enabling efficient and user-friendly experiences.

Automotive Systems:

In the automotive industry, embedded systems are used in systems such as infotainment systems, navigation systems, anti-lock braking systems (ABS), and engine control units (ECUs). They help improve safety, enhance vehicle performance, and increase comfort by controlling critical functions in real time.

Healthcare:
Embedded systems are essential in medical devices such as pacemakers, insulin pumps, patient monitoring systems, and diagnostic equipment. These systems ensure real-time data collection, monitoring, and analysis, which is vital for effective treatment and patient care.

Industrial Automation:

Embedded systems are used in industrial applications for robotics, manufacturing equipment control, temperature regulation, and inventory management systems. They increase efficiency, productivity, and safety in factories and production lines.

Telecommunication:
Embedded systems are at the heart of telecommunications infrastructure, including cellular networks, GPS systems, and satellite communications. They facilitate the real-time transfer of data and enable communication between devices globally.

HomeAppliances:
Everyday household devices, such as washing machines, microwave ovens, refrigerators, and smart thermostats, are powered by embedded systems. These systems automate functions like temperature regulation, timing, and efficiency monitoring.

IoT:
Embedded systems are fundamental to the functioning of IoT devices. These systems are embedded in objects like smartwatches, smart thermostats, and wearable health devices, allowing them to collect and transmit data for analysis and decision-making.

These applications demonstrate the versatility of embedded systems across a wide range of sectors, highlighting their critical role in improving functionality, performance, and automation in modern technology.

2.4 WHY EMBEDDED?

An embedded system is a specialize computer system —a combination of a computer processor, computer memory and input/output peripheral devices—that has a dedicated function within a larger mechanical or electronic system. It is embedded as part of a complete device often including electrical or electronic hardware and mechanical parts. Because an embedded system typically controls physical operations of the machine that it is embedded within, it often has real-time computing constraints. Embedded systems control many devices in common use. In 2009, it was estimated that ninety-eight percent of all microprocessors manufactured were used in embedded systems.

Modern embedded systems are often based on microcontrollers (i.e. microprocessors with integrated memory and peripheral interfaces), but ordinary microprocessors (using external chips for memory and peripheral interface circuits) are also common, especially in more complex systems. In either case, the processor(s) used may be types ranging from general purpose to those specialized in a certain class of computations, or even custom designed for the application at hand. A common standard class of dedicated processors is the digital signal processor (DSP).

Since the embedded system is dedicated to specific tasks, design engineers can optimize it to reduce the size and cost of the product and increase its reliability and performance. Some embedded systems are mass-produced, benefiting from economies of scale.

Embedded systems range in size from portable personal devices such as digital watches and MP3-players to bigger machines like home appliances, industrial assembly lines, robots, transport vehicles, traffic light controllers, and medical imaging systems. Often they constitute subsystems of other machines like avionics in aircraft and astrionics in spacecrafts. Large installations like factories, pipelines and electrical grids rely on multiple embedded systems networ-ked together. Generalized through software customization, embedded systems such as programmable logic controllers frequently comprise their functional units.

The embedded system life cycle involves several key stages, starting with conceptualization, where requirements are defined, followed by system design to determine hardware and software components. Hardware design focuses on creating the physical components, while software development involves programming the system's firmware. Next, integration and testing ensure hardware and software function together, followed by prototyping and validation to test the system under real-world conditions. After refining the design, the system enters manufacturing for mass production and deployment for installation and field testing. Ongoing maintenance and support ensure the system remains functional, and eventually, the system reaches end of life (EOL) when it is retired and replaced.

During hardware design, physical components such as microcontrollers, sensors, and circuits are selected and developed. Software development involves coding the firmware and application software, which is then integrated and tested to ensure all components function as intended. After integration, prototyping and validation ensure the system performs under real-world conditions, leading to production and manufacturing, where the system is mass-produced.

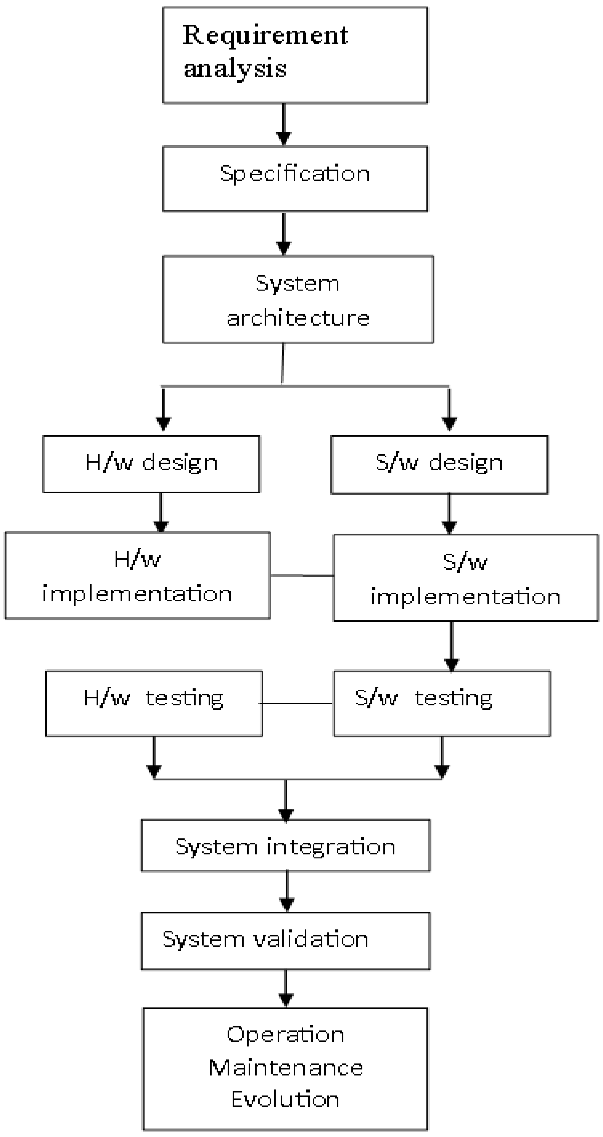


Fig 2.2: Embedded Development Life Cycle

Embedded systems range from those low in complexity, with a single microcontroller chip, to very high with multiple units, peripherals and networks, which may reside in equipment racks or across large geographical areas connected via long-distance communications lines.

2.5 DESIGN APPROACHES

In designing a child safety wearable device, the integration of multiple technologies is key to ensuring its effectiveness, reliability, and user-friendliness. The literature survey on existing child safety wearable systems highlights various design approaches based on functionality, power consumption, communication protocols, and sensor integration. Below is an overview of the design approaches that have been explored in literature for wearable devices targeting child safety, with a focus on the GSM module as a communication interface.

2.5.1 Embedded System With Arduino

A wearable device for child safety is a technological solution designed to provide peace of mind to parents by ensuring the well-being of their children in real-time. The device typically integrates a range of sensors and communication modules to monitor the child’s location, health, and safety status. One of the core features is GPS tracking, which allows parents to monitor their child’s location continuously. This functionality ensures that parents are notified if their child strays outside a designated safe zone, often referred to as a geo-fence. Another essential feature is fall detection, achieved using accelerometers or gyroscopes, which can detect sudden movements or a fall. If a fall is detected, the device can immediately alert the parent through vibration, sound, or an automatic message sent via Bluetooth or GSM.

Health monitoring is another important aspect, where sensors like heart rate monitors and temperature sensors track the child’s vital signs. This data helps ensure the child is not at risk of any health issues. For example, if the child’s heart rate exceeds normal levels or their body temperature is abnormally high or low, an alert is sent to the parent. The device can also include an emergency button, which allows the child to send a distress signal when in danger, providing an immediate communication channel to parents. Additionally, the device is designed to be wearable, often in the form of a wristband or clip, making it both comfortable and convenient for the child to wear. The communication can be achieved via Bluetooth to a smartphone or through a GSM module for SMS alerts. Overall, this wearable device enhances child safety by providing a reliable, real-time monitoring system that alerts parents to potential dangers and health concerns.

2.5.2 GPS Module For Real-Time Location Tracking

The GPS module is integrated into the system to provide real-time tracking of the vehicle’s location. The GPS module sends continuous data regarding the vehicle's position (latitude, longitude, and altitude) to the Arduino microcontroller.

- **Location Data Acquisition**: The GPS module, often based on u-blox or Neo-6M technology, is configured to constantly collect location data at regular intervals (e.g., every 5 seconds or 1 minute). The Arduino processes this data and sends it through the GPS modem to the user for real-time vehicle tracking.

- **Geofencing**: The system can use geofencing features, where the user can set a virtual boundary around a specific location (e.g., a city or parking area). If the vehicle crosses this boundary, the Arduino sends an alert via the GPS modem, notifying the vehicle owner of unauthorized movement.

2.5.3 GSM As a Communication Interface:

The GSM module is widely used in child safety devices for its simple yet effective communication capabilities. GSM provides the ability to send SMS alerts in emergencies, making it a critical part of child safety wearables.

- **SMS Alerts for Emergency Situations**: One of the most commonly discussed uses of GSM in wearable child safety devices is the SMS alert system. In many existing designs, if a child presses the emergency button or experiences a fall, the GSM module sends an SMS to the parent’s phone, indicating the emergency situation and including the child’s location.
- **Geo-fencing**: Another significant design feature discussed in the literature is geo-fencing. In this system, the wearable device is programmed with a geographical boundary (safe zone). When the child exits this boundary, the GSM module sends an immediate SMS alert to the parent, providing the child’s GPS location.
- **Voice Communication**: Some advanced systems also integrate the GSM module for voice calls, allowing the child to make emergency calls directly from the wearable device, or enabling the parent to call and listen in on the surroundings for better situational awareness.

2.5.4 Power Efficiency in Wearable Devices:

One of the biggest challenges identified in the literature is power management. Since wearable devices must be continuously functional, they need efficient power management to extend battery life. The GSM module, known for high power consumption, requires careful consideration.

- **Low Power Consumption Modes**: Researchers recommend the use of low-power microcontrollers and sleep modes for the GSM module and other components (like GPS) when not in use. For example, Arduino-based systems often use power-efficient microcontrollers (like the ATmega328P) and implement sleep modes when the GSM module is idle.
- **Battery Selection**: Rechargeable Li-Po or Li-ion batteries are typically used in wearable devices. The battery capacity must be sufficient to support the high-power GSM module without compromising the device's form factor. Literature suggests using batteries with a higher mAh rating (e.g., 1000mAh or more) for extended operational time.
- **Energy Harvesting**: Some designs incorporate solar panels or other energy-harvesting technologies to supplement battery life, ensuring the device can operate continuously without frequent recharging.

2.5.5 User-Centric Design and Comfort:

Another key consideration highlighted in the literature is user-centric design, especially in wearable safety devices for children. Comfort and usability are essential since the child will wear the device for extended periods.

- **Form Factor**: The device should be lightweight, compact, and durable. Common form factors include wristbands, clips, or patches that can be easily attached to the child's clothing or body.
- **Emergency Features**: The device should include simple, child-friendly features such as an SOS button for emergency situations, which should be easy for the child to activate when needed. Literature suggests that visual or tactile feedback should be provided to the child after pressing the button, indicating the message has been sent successfully.
- **Water Resistance**: Many wearable child safety devices are designed to be waterproof or water-resistant, ensuring that the device remains functional even in

wet conditions, such as rain or during playtime in the pool.

2.5.6 Mobile Application Integration:

The integration of mobile applications with the wearable device is a common approach. The mobile app allows parents to monitor their child's location, health status, and receive alerts in real-time.

- **Mobile App Features**: These apps can display the child's current location, health data, and status of the safety system. The app can also allow the parent to remotely communicate with the wearable device, check on the child's condition, and receive notifications in case of emergencies.
- **Data Privacy and Security**: The literature stresses the importance of implementing encryption protocols and secure communication channels to ensure that sensitive data (e.g., location and health information) is transmitted securely between the wearable device and the parent's mobile app.

2.6 COMBINATION OF LOGIC DEVICES

The combination of logic devices in the context of an embedded system, especially for a child safety wearable device, refers to integrating multiple components such as microcontrollers, sensors, communication modules, and actuators to form a cohesive system. Logic devices enable the processing, control, and decision-making within the system. The logical combination of these devices allows the system to perform complex tasks like monitoring the child's health, sending alerts, and responding to emergency situations.

1. Location Tracking and Geofencing:

GPS Module: Use a GPS module to track the child's location. The GPS module communicates with the microcontroller to get real-time location coordinates (latitude and longitude).

Geofencing: Set a virtual boundary around the child's location. If the child moves beyond the designated boundary, the device will send an alert to the parent's smartphone or a predefined number.

2. SOS Button for Emergency Alerts:

Implement an emergency button (or accelerometer-based detection) that the child can press if they need help. Pressing the button will send an emergency signal (SMS, app notification, or automated call) to the parent or guardian’s phone. An LED indicator can be used to signal whether the SOS feature is active or not.

3. Fall Detection:

Use an accelerometer and gyroscope to detect sudden movements or falls. When abnormal movements are detected (such as a fall), the device will alert the parent through a mobile app or send an automated emergency message. The sensor data from the accelerometer will be processed to detect motion patterns that resemble a fall or an accident.

4. Battery and Power Management:

Use low-power components to maximize battery life, and implement an efficient power management system. The device should be able to run for several hours or days without frequent charging. Rechargeable Lithium Polymer (Li-Po) batteries are commonly used for wearable devices.

5. Mobile Application:

A mobile app (for Android or iOS) can be developed to receive real-time updates about the child’s location, health, and emergency alerts. The app could allow the parent to. Track the child’s location in real-time. Set geofence boundaries. Receive push notifications if the child moves out of the designated zone. Enable two-way communication. Get alerts if the child falls or if the SOS button is pressed.

CHAPTER-3
HARDWARE REQUIREMETS

3.1 HARDWARE

Overview The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter. Revision 2 of the Uno board has a resistor pulling the 8U2 HWB line to ground, making it easier to put into DFU mode. Revision 3 of the board has the following new features:

- pinout: added SDA and SCL pins that are near to the AREF pin and two other new pins placed near to the RESET pin, the IOREF that allow the shields to adapt to the voltage provided from the board. In future, shields will be compatible both with the board that use the AVR, which operate with 5V and with the Arduino Due that operate with 3.3V. The second one is a not connected pin, that is reserved for future purposes.
- Stronger RESET circuit.
- Atmega 16U2 replace the 8U2. "Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version 1.0 will be the reference versions of Arduino, moving forward. The Uno is the latest in a series of USB Arduino boards, and the reference model for the Arduino platform; for a comparison with previous versions, see the index of Arduino boards.
- Microcontroller ATmega328
- Operating Voltage 5V
- Input Voltage (recommended) 7-12V
- Input Voltage (limits) 6-20V
- Digital I/O Pins 14 (of which 6 provide PWM output)
- Analog Input Pins 6
- DC Current per I/O Pin 40 mA
- DC Current for 3.3V Pin 50 mA
- Flash Memory 32 KB (ATmega328) of which 0.5 KB used by bootloader

- SRAM 2 KB (ATmega328)
- EEPROM 1 KB (ATmega328)
- Clock Speed 16 MHz

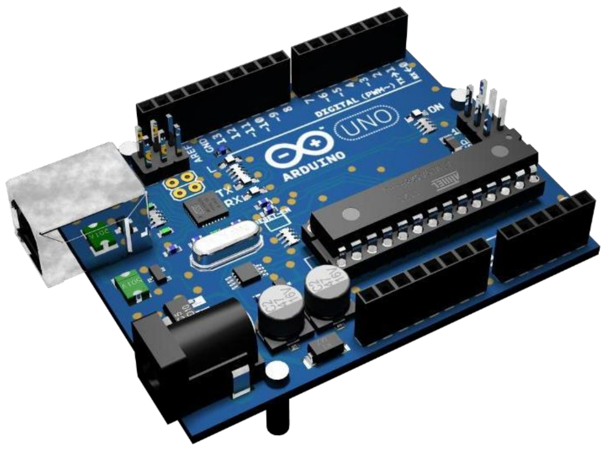


Fig 3.1: Arduino board

Schematic & Reference Design

EAGLE files: arduino-uno-Rev3-reference-design.zip (NOTE: works with Eagle 6.0 and newer) Schematic: arduino-uno-Rev3-schematic.pdf Note: The Arduino reference design can use an Atmega8, 168, or 328, Current models use an ATmega328, but an Atmega8 is shown in the schematic for reference. The pin configuration is identical on all three processors.

Power

The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm centre-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The power pins are as follows:

VIN: The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin. 5V. This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board. We don't advise it. 3V3. A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.

GND: Ground pins.

Memory:

The ATmega328 has 32 KB (with 0.5 KB used for the bootloader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

Input and Output:

Each of the 14 digital pins on the Uno can be used as an input or output, using pinMode(), digital Write(), and digital Read() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kohms. In addition, some pins have specialized functions:

- Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
- External Interrupts: 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analogWrite() function.
- SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.
- LED: 13. There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off. The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function. Additionally, some pins have specialized functionality:
- TWI: A4 or SDA pin and A5 or SCL pin. Support TWI communication using the Wire library. There are a couple of other pins on the board:

- AREF.Reference voltage for the analog inputs. Used with analogReference().
- Reset. Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board. See also the mapping between Arduino pins and ATmega328 ports. The mapping for the Atmega8, 168, and 328 is identical.

Communication:

The Arduino Uno has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a .inf file is required. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1). A SoftwareSerial library allows for serial communication on any of the Uno's digital pins. The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation for details.

3.1.1 Power Supply

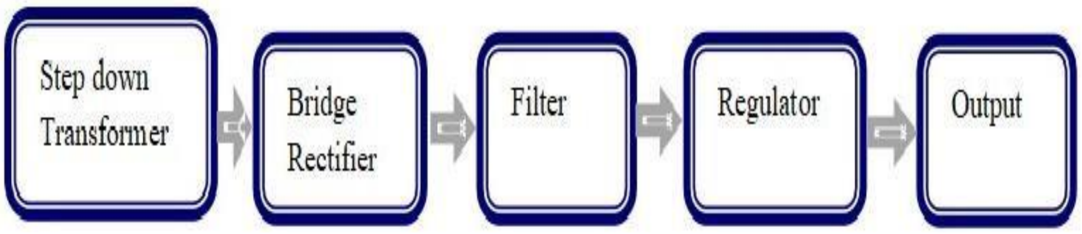


Fig 3.2: Block diagram for power supply

The input to the circuit is applied from the regulated power supply. The a.c. input i.e., 230V from the mains supply is step down by the transformer to 12V and is fed to a rectifier. The output obtained from the rectifier is a pulsating d.c voltage.

In order to get a pure d.c voltage, the output voltage from the rectifier is fed to a filter to remove any a.c components present even after rectification. Now, this voltage is given to a voltage regulator to obtain a pure constant dc voltage. The design of the power supply must account for factors such as power consumption, voltage regulation, and energy efficiency, as many embedded systems operate in environments where power resources are limited, such as in battery-powered or remote applications.

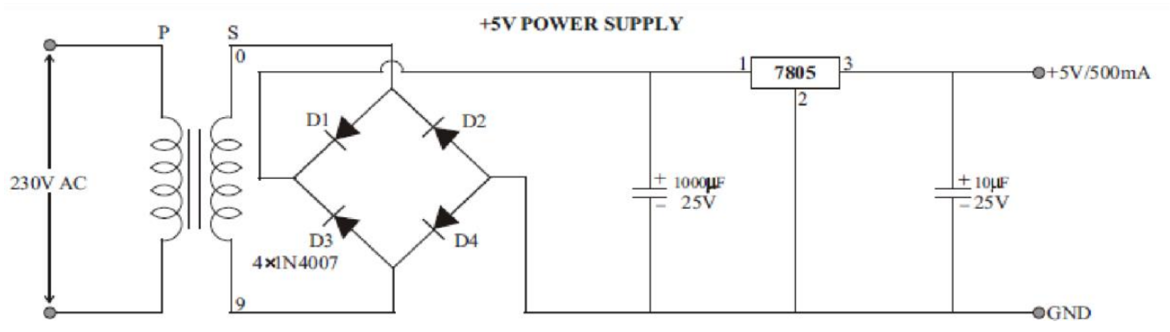


Fig 3.3: Circuit diagram of power supply

3.1.2 Step Down Transformer

Usually, DC voltages are required to operate various electronic equipment and these voltages are 5V, 9V or 12V. But these voltages cannot be obtained directly. Thus the a.c input available at the mains supply i.e., 230V is to be brought down to the required voltage level. This is done by a transformer. Thus, a step down transformer is employed to decrease the voltage to a required level.

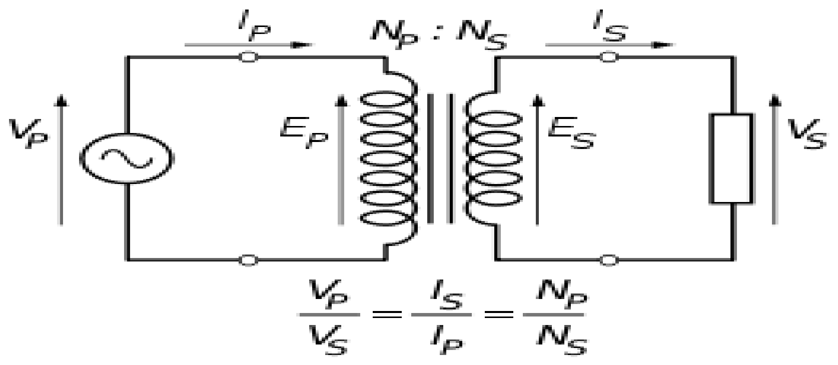


Fig 3.4: Step-down transformer

3.1.3Rectifier:

Input **Output**

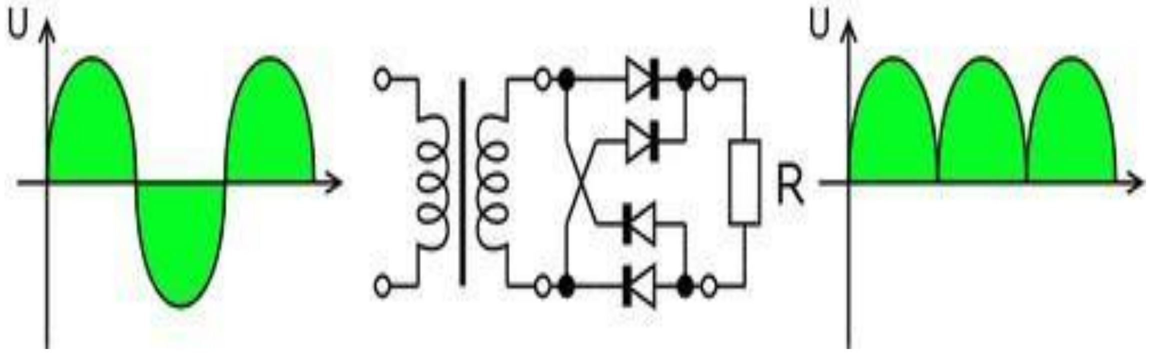


Fig 3.5: Bridge rectifier

The output from the transformer is fed to the rectifier. It converts A.C. into pulsating D.C. The rectifier may be a half wave or a full wave rectifier. In this project, a bridge rectifier is used because of its merits like good stability and full wave rectification.

3.1.4 Filter

Capacitive filter is used in this project. It removes the ripples from the output of rectifier and smoothens the D.C. Output received from this filter is constant until the mains voltage and load is maintained constant. However, if either of the two is varied, D.C. voltage received at this point changes. Therefore a regulator is applied at the output stage.

3.1.5 Voltage Regulator

As the name itself implies, it regulates the input applied to it. A voltage regulator is an electrical regulator designed to automatically maintain a constant voltage level. In this project, power supply of 5V and 12V are required. In order to obtain these voltage levels, 7805 and 7812 voltage regulators are to be used. The first number 78 represents positive supply and the numbers 05, 12 represent the required output voltage levels.

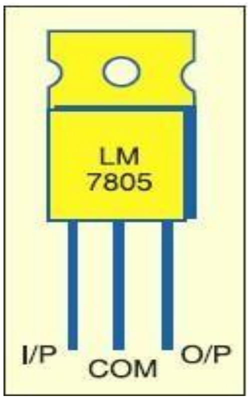


Fig 3.6: Voltage Regulator

Features:

- Output Current up to 1A.
- Output Voltages of 5, 6, 8, 9, 10, 12, 15, 18, 24V.
- Thermal Overload Protection.
- Short Circuit Protection.
- Output Transistor Safe Operating Area Protection.

3.1.6 LCD/Display

Liquid Crystal Display also called as LCD is very helpful in providing user interface as well as for debugging purpose. The most commonly used Character based LCDs are based on Hitachi's HD44780 controller or other which are compatible with HD44580.

The most commonly used LCDs found in the market today are 1 Line, 2 Line or 4 Line LCDs which have only 1 controller and support at most of 80 characters, whereas LCDs supporting more than 80 characters make use of 2 HD44780 controllers.

Pin Description:

Table 3.1.1:Pin description of LCD

Pin No.	Name	Description
1	**VSS**	Power supply (GND)
2	**VCC**	Power supply (+5V)
3	**VEE**	Contrast adjust
4	**RS**	0=Instruction input 1= Data input
5	**R/W**	0=WritetoLCDmodule 1 = Read from LCD module
6	**EN**	Enable signal
7	**D0**	Data bus line 0 (LSB)
8	**D1**	Data bus line 1
9	**D2**	Data bus line 2
10	**D3**	Data bus line 3
11	**D4**	Data bus line 4
12	**D5**	Data bus line 5
13	**D6**	Data bus line 6
14	**D7**	Data bus line 7 (MSB)
15	**LED+**	Back Light VCC
16	**LED-**	Back Light GND

Although looking at the table you can make your own commands and test them. Below is a brief list of useful commands which are used frequently while working on the LCD.

Table 3.1.2:Command List of LCD display

No.	Instruction	Hex	Decimal
1	Function Set: 8-bit, 1 Line, 5x7 Dots	0x30	48
2	Function Set: 8-bit, 2 Line, 5x7 Dots	0x38	56
3	Function Set: 4-bit, 1 Line, 5x7 Dots	0x20	32
4	Function Set: 4-bit, 2 Line, 5x7 Dots	0x28	40

5	Entry Mode	0x06	6
6	DisplayoffCursoroff (clearing display without clearing DDRAM content)	0x08	8
7	Display on Cursor on	0x0E	14
8	Display on Cursor off	0x0C	12
9	Display on Cursor blinking	0x0F	15
10	Shift entire display left	0x18	24
12	Shift entire display right	0x1C	30
13	Move cursor left by one character	0x10	16
14	Move cursor right by one character	0x14	20
15	Clear Display (also clear DDRAM content)	0x01	1
16	Set DDRAM address or cursor position on display	0x80+add	128+add
17	Set CGRAM address or set pointer to CGRAM location	0x40+add	64+add

Sending Commands to LCD

To send commands we simply need to select the command register. Everything is same as we have done in the initialization routine. But we will summarize the common steps and put them in a single subroutine. Following are the steps:

- move data to LCD port
- select command register
- select write operation
- send enable signal
- wait for LCD to process the command
- Sending Data to LCD
- To send data move data to LCD port

DISPLAY:

Fig 3.7: 2x16 LCD Display

- often called a notebook, is a small, portable personal computer (PC) with a "clamshell" form factor, typically having a thin LCD or LED computer screen mounted on the inside of the upper lid of the clamshell and an alphaumeric keyboard on the inside of the lower lid. The clamshell is opened up to use the computer. Laptops are folded shut for transportation, and thus are suitable for mobile use. Its name comes from lap, as it was deemed to be placed on a person's lap when being used. Although originally there was a distinction between laptops and notebooks (the former being bigger and heavier than the latter), as of 2014, there is often no longer any difference. Today, laptops are commonly used in a variety of settings, such as at work, in education, for playing games, Internet surfing, for personal multimedia, and general home computer use.
- Laptops combine all the input/output components and capabilities of a desktop computer, including the display screen, small speakers, a keyboard, data storage device, optical disc drive, pointing devices (such as a touch pad or trackpad), a processor, and memory into a single unit. Most modern laptops feature integrated webcams and built-in microphones, while many also have touchscreens. Laptops can

  be powered either from an internal battery or by an external power supply from an AC adapter. Hardware specifications, such as the processor speed and memory capacity, significantly vary between different types, makes, models and price points.
- Design elements, form factor and construction can also vary significantly between models depending on intended use. Examples of specialized models of laptops include rugged notebooks for use in construction or military appliances, as well as low production cost laptops such as those from the One Laptop per Child (OLPC) organization, which incorporate features like solar charging and semi-flexible components not found on most laptop computers. Portable computers, which later developed into modern laptops, were originally considered to be a small niche market, mostly for specialized field applications, such as in the military, for accountants, or for traveling sales representatives. As the portable computers evolved into the modern laptop, they became widely used for a variety of purposes.

3.1.7 GPS Module

GPS technology is increasingly being used in wearable devices designed for child safety. By incorporating GPS into wristbands, watches, or other wearable items, parents can track their child's real-time location and ensure their safety. These devices typically use GPS satellites to determine the child's position, which is then sent to a smartphone or another device via GSM or Wi-Fi connectivity. Features like geo-fencing allow parents to set safe zones, sending alerts if the child leaves these predefined areas, ensuring they are always within a designated, safe boundary. Additionally, many GPS-enabled child safety wearables include emergency features like an SOS button, which sends immediate alerts to parents in case of danger or when the child needs assistance. Such devices provide peace of mind for parents, offering real-time tracking and immediate communication, which is especially helpful in crowded places or unfamiliar environments. GPS tracking for child safety not only provides a reliable way to locate a child but also ensures a faster response in emergencies.

PIN DESCRIPTION:

VCC: Power Supply 3.3 – 6 V

GND: Ground

TX: Transmit data serially which gives information about location, time etc.

RX: Receive Data serially. It is required when we want to configure GPS module.

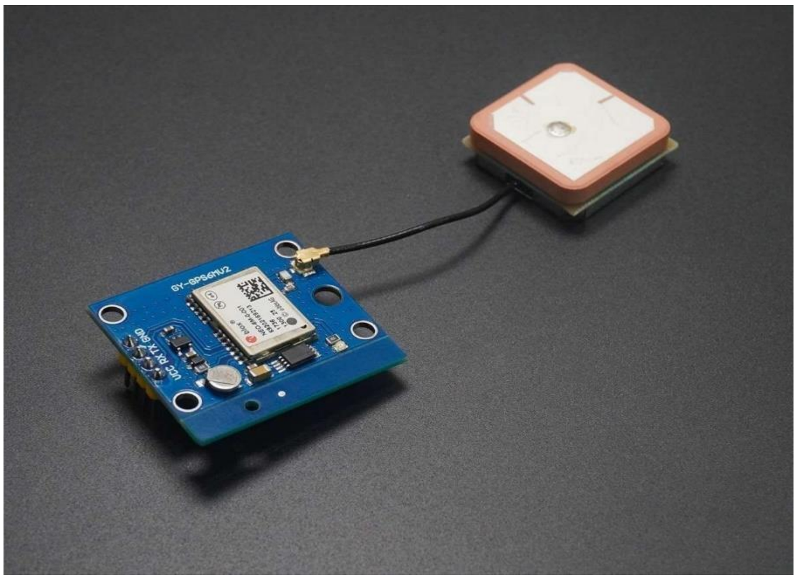


Fig 3.8: GPS Module

3.1.8 GSM Module

The GSM module plays a crucial role in child safety wearables by enabling real-time communication between the child's wearable device and the parent’s phone. This module uses the GSM (Global System for Mobile Communications) network to send SMS alerts, make calls, or transmit location information when needed. In a child safety device, the GSM module can be programmed to send an SOS message to a parent's phone if the child is in distress or has pressed an emergency button. It can also transmit the child's GPS coordinates, allowing parents to track their child's exact location. The GSM module is powered by a SIM card, which connects the device to the mobile network, ensuring reliable communication even when GPS signals might be weak.

Communication: It allows SMS, voice calls, and GPRS (data transmission) over a cellular network.

GSM Bands: Supports GSM bands like 850/900 MHz, making it ideal for use in regions with these frequencies.

Power Requirements: Operates with 3.4V to 4.4V of power. It requires a stable power supply.

Pin Interface:

- TXD (Transmit),
- RXD (Receive) for serial communication with microcontrollers.
- RESET to restart the module.
- SIM card interface for mobile network connectivity.

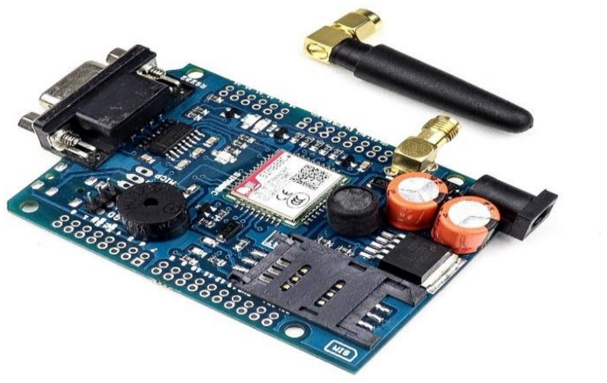


Fig 3.9: GSM Module

3.1.9 MEMS Sensor

MEMS (Micro-Electro-Mechanical Systems) play an important role in enhancing child safety through their small, sensitive, and low-power sensors that can be integrated into wearable devices. For example, MEMS accelerometers and gyroscopes can be used in smartwatches or bands to monitor a child's movement and detect abnormal behavior, such as falls or sudden changes in position. These sensors can trigger an alert to parents or guardians if a fall or accident occurs. Additionally, MEMS pressure sensors can be incorporated to monitor environmental conditions, like detecting unusual pressure changes in the surroundings, which could indicate an emergency situation. These compact and energy-efficient sensors are ideal for child safety wearables, offering real-time data and alerts to ensure the child's well-being in various scenarios, such as outdoor play or in crowded areas. MEMS technology, with its small form factor and high accuracy, provides a reliable solution for keeping children safe in modern, connected environments.

MEMS Sensor Pin Description

VCC - Powers the sensor (usually 3.3V or 5V).

GND - Ground pin.

SCL - Clock pin for I2C.

SDA - Data pin for I2C.

INT - Interrupt pin for triggering an alert.

AD0 - I2C address pin

Fig 3.9: GSM Module

CHAPTER-4

SOFTWARE REQUIREMENTS

4.1 ARDUINO SOFTWARE

The Arduino is a family of microcontroller boards to simplify electronic design, prototyping and experimenting for artists, hackers, hobbyists, but also many professionals. People use it as brains for their robots, to build new digital music instruments, or to build a system that lets your house plants tweet you when they're dry. Arduinos (we use the standard Arduino Uno) are built around an ATmega microcontroller — essentially a complete computer with CPU, RAM, Flash memory, and input/output.

What you will need:

- A computer (Windows, Mac, or Linux)
- An Arduino-compatible microcontroller (anything from this guide should work)
- A USB A-to-B cable, or another appropriate way to connect your Arduinocompatible microcontroller to your computer (check out this USB buying guide if you're not sure which cable to get).

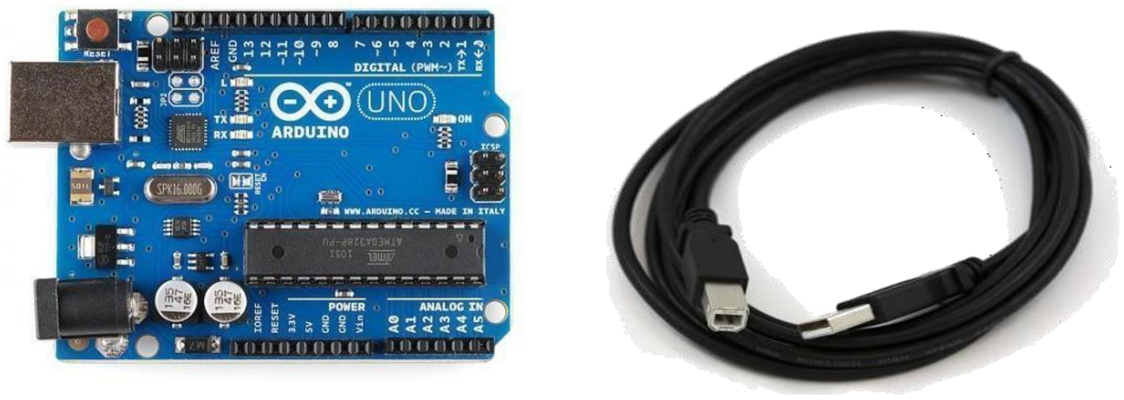


Fig 4.1: Arduino UNO

- An Arduino Uno
- Windows 7, Vista, and XP
- Installing the Drivers for the Arduino Uno (from Arduino.cc)
- Plug in your board and wait for Windows to begin it's driver installation process After a few moments, the process will fail, despite its best efforts
- Click on the Start Menu, and open up the Control Panel
- Look under Ports (COM & LPT). You should see an open port named "Arduino UNO (COMxx)".

- While in the Control Panel, navigate to System and Security. Next, click on System Once the System window is up, open the Device Manager.

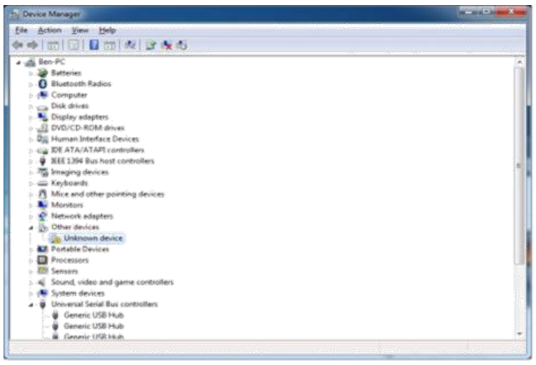
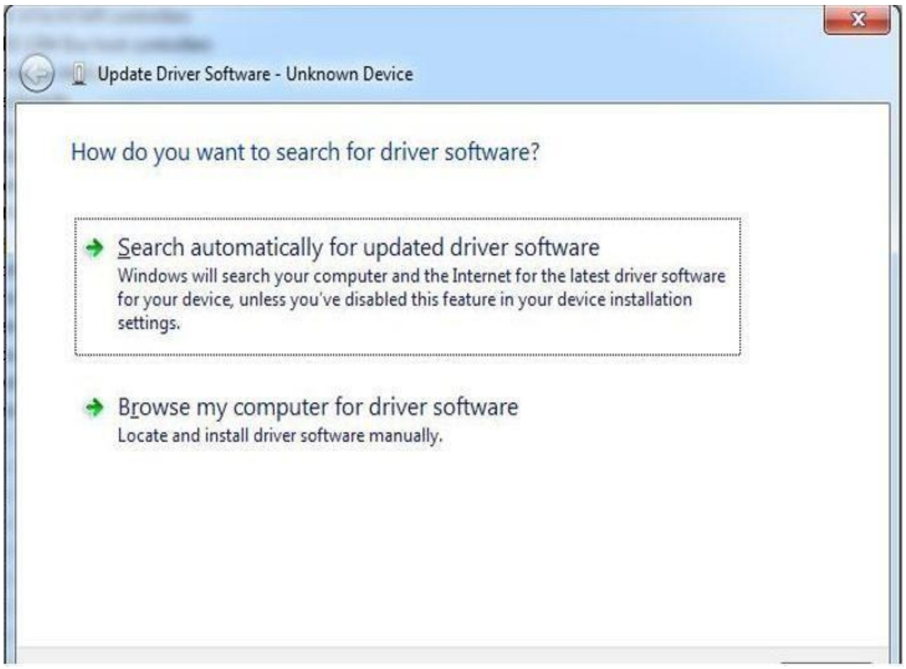


Fig 4.2: Device Manager

- If there is no COM & LPT section, look under ‘Other Devices’ for ‘Unknown Device’.

- Right click on the “Arduino UNO (COMxx)” or “Unknown Device” port and choose the “Update Driver Software” opti Next, choose the “Browse my computer for Driver software” option.



How do you want to search for driver software?

Search automatically for updated driver software
Windows will search your computer and the Internet for the latest driver software for your device, unless you've disabled this feature in your device installation settings.

Browse my computer for driver software
Locate and install driver software manually.

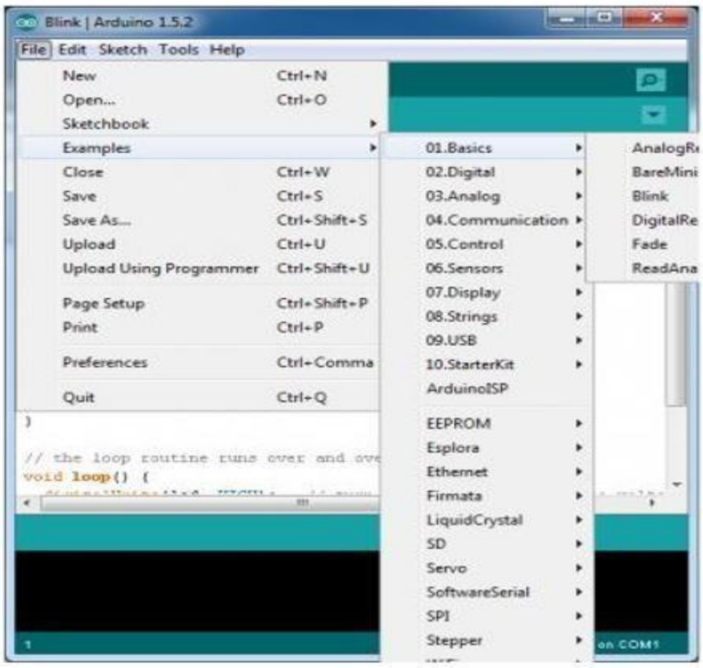
Fig 4.3: Update Driver Software

- Finally, navigate to and select the Uno’s driver file, named “ArduinoUNO.inf”, located in the “Drivers” folder of the Arduino Software download (not the “FTDI USB Drivers” sub-directory).

- If you cannot see the .inf file, it is probably just hidden. You can select the ‘drivers’ folder with the ‘search sub-folders’ option selected instead. ○
Windows will finish up the driver installation.

After following the appropriate steps for your software install, we are now ready to test your first program with your Arduino board!

- Launch the Arduino application
- If you disconnected your board, plug it back in
- Open the Blink example sketch by going to: File > Examples > 1.Basics > Blink
- After a second, you should see some LEDs flashing on your Arduino, followed by the message 'Done Uploading' in the status bar of the Blink sketch.

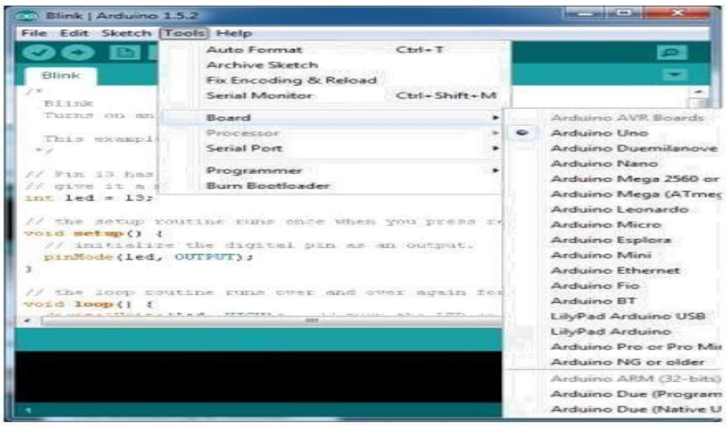


```
}

// the loop routine runs over and ove
void loop() {
```

Fig 4.4: Arduino File basics

- If everything worked, the onboard LED on your Arduino should now be blinking! You just programmed your first Arduino!
- Select the type of Arduino board you're using: Tools > Board > your board type

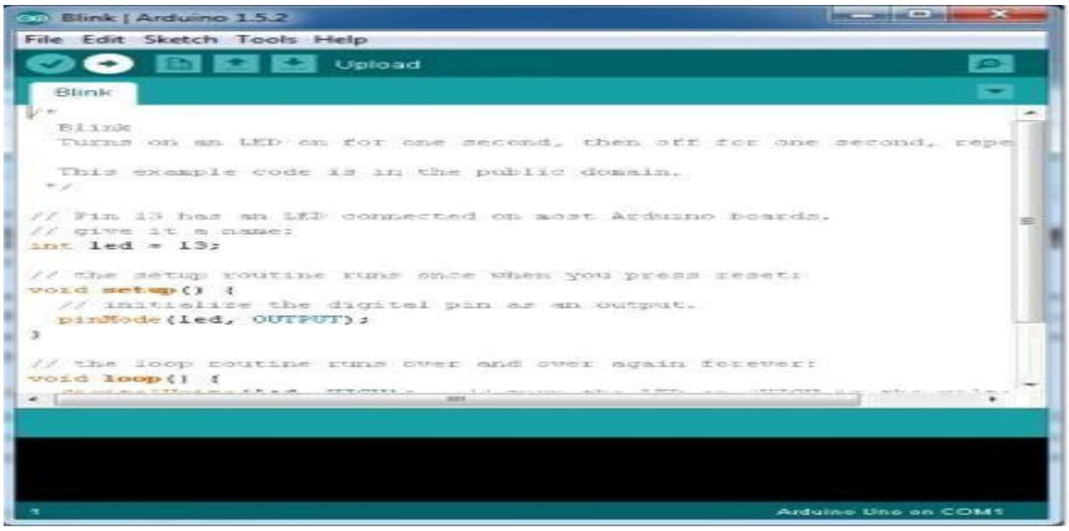


```
/*
  Blink
  Turns on an
  This exampl
 */
// Pin 13 has
// give it a
int led = 13;
// the setup routine runs once when you press r
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}
// the loop routine runs over and over again fo
void loop() {
```

Fig 4.5: Arduino Tools Board

- Select the serial/COM port that your Arduino is attached to: Tools > Port > COMxx

- If you’re not sure which serial device is your Arduino, take a look at the available ports, then unplug your Arduino and look again.
- The one that disappeared is your Arduino.With your Arduino board connected, and the Blink sketch open, press the ‘Upload’ button.



```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repe

  This example code is in the public domain.
 */

// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
```

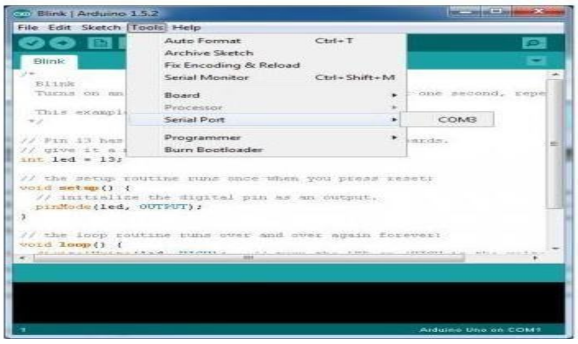


Fig 4.6: Blink Arduino Tools

CHAPTER -5

WORKING MODEL AND ITS COMPONENTS

5.1 BLOCK DIAGRAM

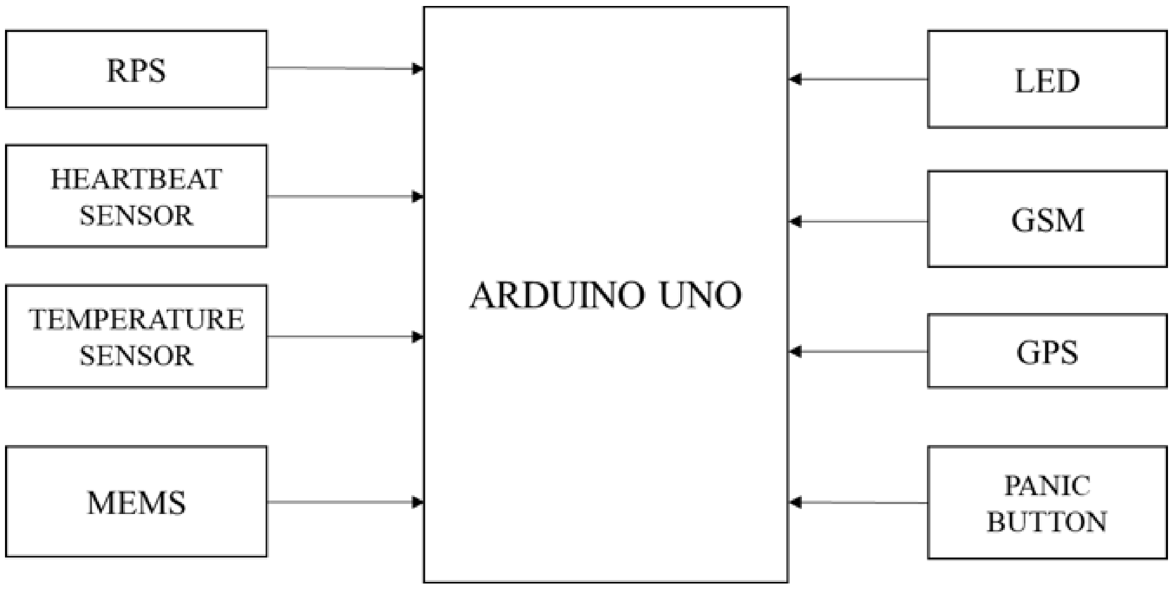


Fig 5.1: Block Diagram

5.2 WORKING

5.2.1 Introduction To Arduino:

The Arduino is a family of microcontroller boards to simplify electronic design, prototyping and experimenting for artists, hackers, hobbyists, but also many professionals. People use it as brains for their robots, to build new digital music instruments, or to build a system that lets your house plants tweet you when they're dry. Arduinos (we use the standard Arduino Uno) are built around an ATmega microcontroller — essentially a complete computer with CPU, RAM, Flash memory, and input/output pins, all on a single chip. Unlike, say, a Raspberry Pi, it's designed to attach all kinds of sensors, LEDs, small motors and speakers, servos, etc. directly to these pins, which can read in or output digital or analog voltages between 0 and 5 volts.

The Arduino connects to your computer via USB, where you program it in a simple language (C/C++, similar to Java) from inside the free Arduino IDE by uploading your compiled code to the board. Once programmed, the Arduino can run with the USB link back to your computer, or stand-alone without it — no keyboard or screen needed, just power.

Arduino boards are microcontroller-based platforms that enable users to create various electronic projects. The most common board is the Arduino Uno, which features an ATmega328P microcontroller, but there are several other models tailored for specific applications, such as the Arduino Mega, Nano, and Leonardo.

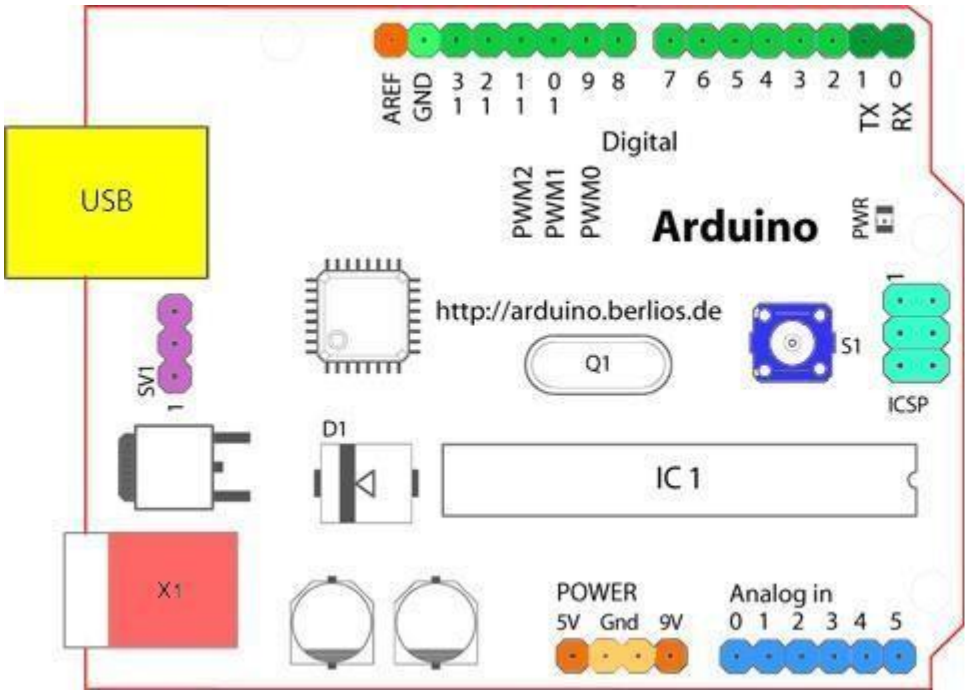


Fig 5.2: Structure of Arduino Board

Looking at the board from the top down, this is an outline of what you will see (parts of the board you might interact with in the course of normal use are highlighted)

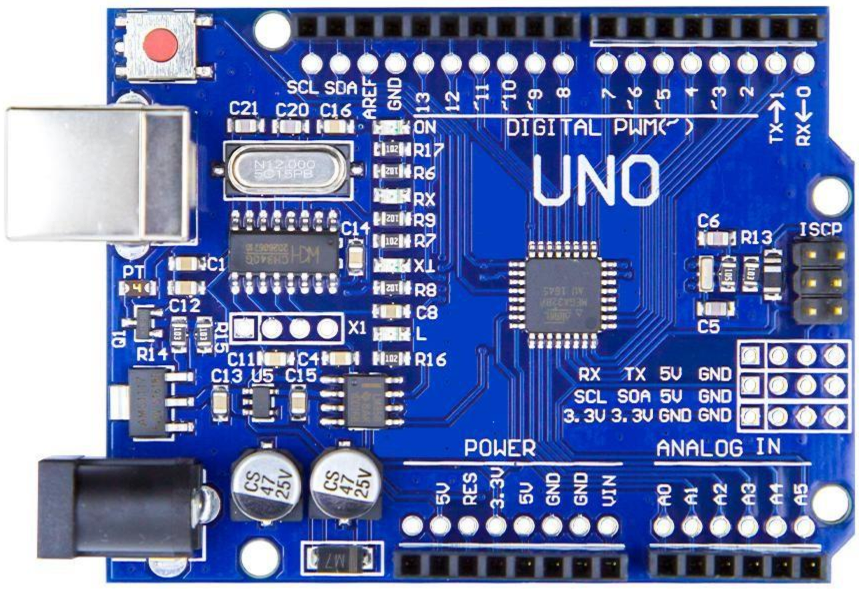


Fig5.3:Arduino Board

Starting clockwise from the top center:

- Analog Reference pin (orange)
- Digital Ground (light green)

- Digital Pins 0-1/Serial In/Out - TX/RX (dark green) - These pins cannot be used for digital i/o (Digital Read and Digital Write) if you are also using serial communication (e.g. Serial.begin).
- Reset Button - S1 (dark blue)
- In-circuit Serial Programmer (blue-green)
- Analog In Pins 0-5 (light blue)
- Power and Ground Pins (power: orange, grounds: light orange)
- External Power Supply In (9-12VDC) - X1 (pink)
- Toggles External Power and USB Power (place jumper on two pins closest to desired supply) - SV1 (purple)
- USB (used for uploading sketches to the board and for serial communication between the board and the computer; can be used to power the board) (yellow)

DIGITAL PINS

In addition to the specific functions listed below, the digital pins on an Arduino board can be used for general purpose input and output via the pin mode(), Digital read(), and Digital write() commands. Each pin has an internal pull-up resistor which can be turned on and off using digital Write() (w/ a value of HIGH or LOW, respectively) when the pin is configured as an input. The maximum current per pin is 40mA.

- **Serial: 0 (RX) and 1 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. On the Arduino Diecimila, these pins are connected to the corresponding pins of the FTDI USB-to-TTL Serial chip. On the Arduino BT, they are connected to the corresponding pins of the WT11 Bluetooth module. On the Arduino Mini and LilyPad Arduino, they are intended for use with an external TTL serial module (e.g. the Mini-USB Adapter).
- **External Interrupts:** 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attach Interrupt(), function for details.
- **PWM:** 3, 5, 6, 9, 10, and 11 Provide 8-bit PWM output with the analog write() function. On boards with an ATmega8, PWM output is available only on pins 9, 10, and 11.
- **BT Reset: 7.** (Arduino BT-only) Connected to the reset line of the Bluetooth module.

- **SPI:** 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language.
- **LED: 13.** On the Decimole and Lilypad, there is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

ANALOG PINS

In addition to the specific functions listed below, the analog input pins support 10-bit analog-to-digital conversion (ADC) using the analog Read() function. Most of the analog inputs can also be used as digital pins: analog input 0 as digital pin 14 through analog input 5 as digital pin 19. Analog inputs 6 and 7 (present on the Mini and BT) cannot be used as digital pins.

- **I^{2}C:** 4 (SDA) and 5 (SCL). Support I^{2}C (TWI) communication using the Wire library (documentation on the Wiring website).

POWER PINS

- **VIN** (sometimes labelled "9V"): The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin. Also note that the Lily Pad has no VIN pin and accepts only a regulated input.
- **5V:** The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.
- **3V3** (Decimole-only) : A 3.3 volt supply generated by the on-board FTDI chip.
- **GND:** Ground pins.

OTHER PINS

- **AREF:** Reference voltage for the analog inputs. Used with analog Reference().
- **Reset:** (decimole-only) Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

ATMEGA328

The ATmega328 is an 8-bit microcontroller based on the AVR architecture. It is popular for its balance of performance, power consumption, and ease of use, making it a favourite among hobbyists and professionals for various electronics projects.

The ATmega328 can be programmed using the Arduino IDE, which simplifies the process with a user-friendly interface and a set of libraries. Users typically write in a simplified version of C/C++. The IDE also provides built-in functions that allow for easy interaction with the microcontroller's hardware features.

PIN DIAGRAM

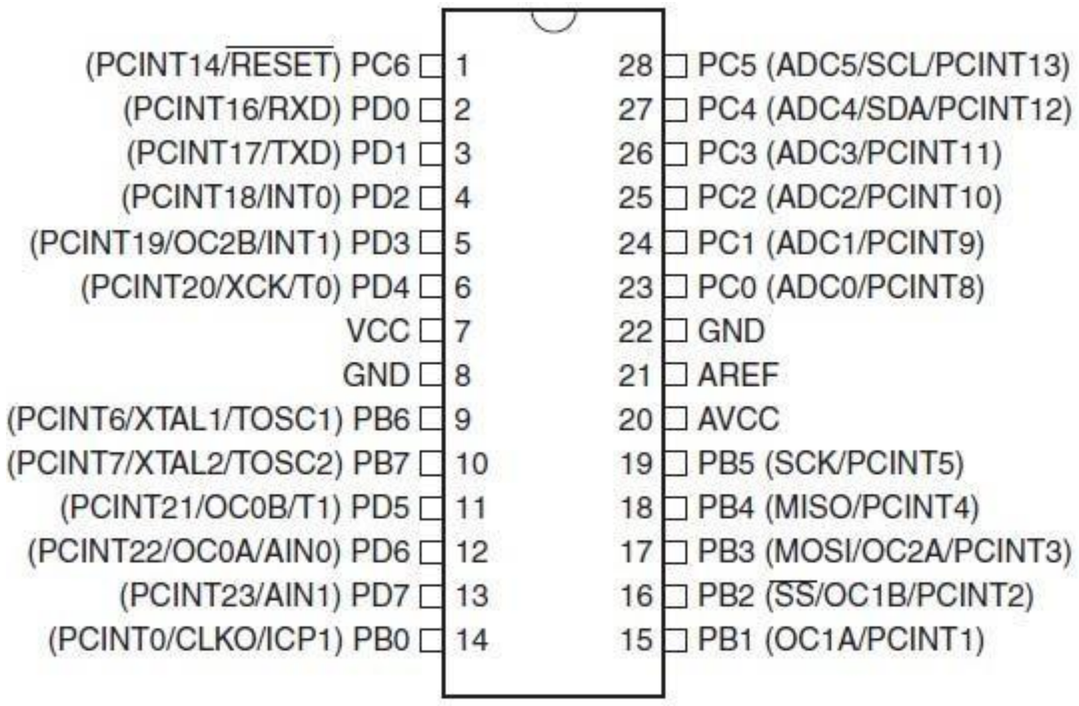


Fig 5.4: Pin Configuration of Atmega328

Pin Description VCC:

Digital supply voltage.

GND:

Ground.

Port A (PA7-PA0):

Port A serves as the analog inputs to the A/D Converter. Port A also serves as an 8-bit bidirectional I/O port, if the A/D Converter is not used. Port pins can provide internal pull-up resistors (selected for each bit). The Port A output buffers have symmetrical drive characteristics with both high sink and source capability. When pins PA0 to PA7 are used as inputs and are externally pulled low, they will source current if the internal pull-up resistors are activated. The Port A pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port B (PB7-PB0):

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running. Port B also serves the functions of various special features of the ATmega32.

Port C (PC7-PC0):

Port C is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port C output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running. If the JTAG interface is enabled, the pull-up resistors on pins PC5(TDI), PC3(TMS) and PC2(TCK) will be activated even if a reset occurs. The TD0 pin is tri-stated unless TAP states that shift out data are entered.

Port D (PD7-PD0):

Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running. Port D also serves the functions of various special features of the ATmega32.

Reset (Reset Input):

A low level on this pin for longer than the minimum pulse length will generate a reset, even if the clock is not running. Shorter pulses are not guaranteed to generate a reset.

XTAL1:

Input to the inverting Oscillator amplifier and input to the internal clock operating circuit.

XTAL2:

Output from the inverting Oscillator amplifier.

AVCC:

AVCC is the supply voltage pin for Port A and the A/D Converter. It should be externally connected to VCC, even if the ADC is not used. If the ADC is used, it should be connected to VCC through a low-pass filter.

AREF:

AREF is the analog reference pin for the A/D Converter.

FEATURES

- 1.8-5.5V operating range
- Up to 20MHz
- Part: ATMEGA328P-AU
- 32kB Flash program memory
- 1kB EEPROM
- 2kB Internal SRAM
- 2 8-bit Timer/Counters
- 16-bit Timer/Counter
- RTC with separate oscillator
- 6 PWM Channels
- 8 Channel 10-bit ADC
- Serial USART
- Master/Slave SPI interface
- 2-wire (I2C) interface
- Watchdog timer
- Analog comparator
- 23 IO lines
- Data retention: 20 years at 85C/ 100 years at 25C
- Digital I/O Pins are 14 (out of which 6 provide PWM output) ○ Analog Input Pins are 6.
- DC Current per I/O is 40 mA
- DC Current for 3.3V Pin is 50mA

AVR CPU CORE

The AVR core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

In order to maximize performance and parallelism, the AVR uses a Harvard architecture with separate memories and buses for program and data. Instructions in the program memory are executed with a single level pipelining. While one instruction is being executed, the next instruction is pre-fetched from the program memory. This concept enables instructions to be executed in every clock cycle. The program memory is In-System Reprogrammable Flash memory. The fast-access Register File contains 32 x 8-bit general purpose working registers with a single clock cycle access time. This allows single-cycle Arithmetic Logic Unit (ALU) operation. In a typical ALU operation, two operands are output from the Register File, the operation is executed, and the result is stored back in the Register File– in one clock cycle.

The main function of the CPU core is to ensure correct program execution. The AVR CPU is capable to access memories, perform calculations, control peripherals, and handle interrupts.

OVERVIEW

This section discusses the AVR core architecture in general. The main function of the CPU core is to ensure correct program execution. The CPU must therefore be able to access memories, perform calculations, control peripherals, and handle interrupts.

Six of the 32 registers can be used as three 16-bit indirect address register pointers for Data Space addressing – enabling efficient address calculations. One of these address pointers can also be used as an address pointer for look up tables in Flash program memory. These added function registers are the 16-bit X-, Y-, and Z-register, described later in this section.

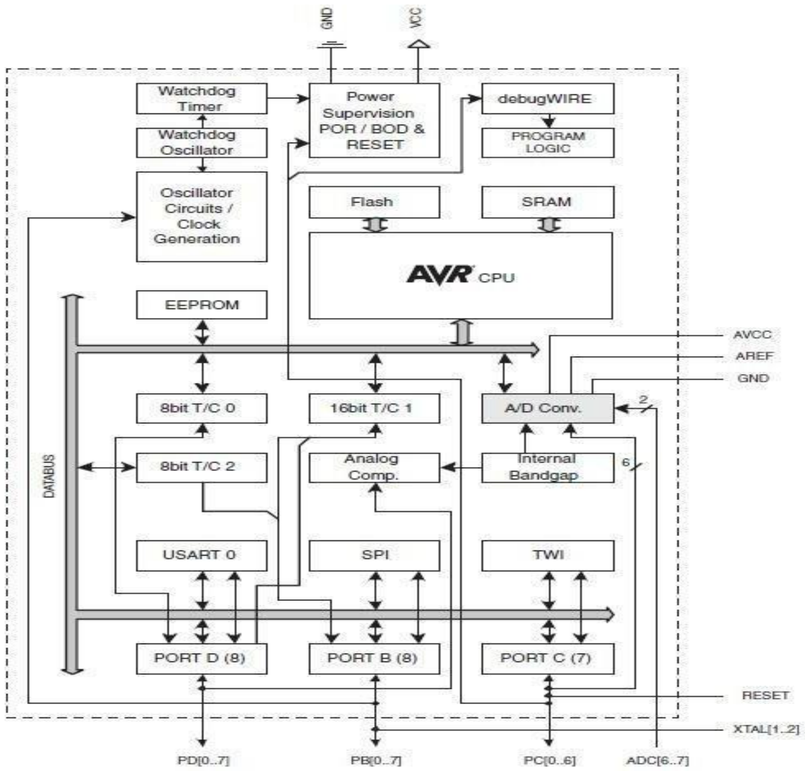


Fig 5.5: Block Diagram

The ALU supports arithmetic and logic operations between registers or between a constant and a register. Single register operations can also be executed in the ALU. After an arithmetic operation, the Status Register is updated to reflect information about the result of the operation. An advanced version of a microcomputer that is integrated into a tiny chip is known as the AVR microcontroller. This microcontroller includes a processor, programmable I/O peripherals & memory. The memory spaces in the AVR architecture are all linear and regular memory maps. A flexible interrupt module has its control registers in the I/O space with an additional Global Interrupt Enable bit in the Status Register. All interrupts have a separate Interrupt Vector in the Interrupt Vector table. The interrupts have priority in accordance with their Interrupt Vector position. The lower the Interrupt Vector address, the higher the priority.

Program flow is provided by conditional and unconditional jump and call instructions, able to directly address the whole address space. Most AVR instructions have a single 16-bit word format. Every program memory address contains a 16- or 32-bit instruction.

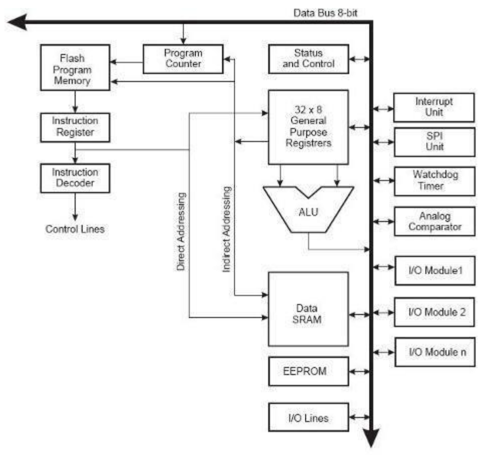


Fig 5.6: AVR core architecture

Program Flash memory space is divided in two sections, the Boot Program section and the Application Program section. Both sections have dedicated Lock bits for write and read/write protection. The SPM instruction that writes into the Application Flash memory section must reside in the Boot Program section. During interrupts and subroutine calls, the return address Program Counter (PC) is stored on the Stack. The Stack is effectively allocated in the general data SRAM, and consequently the Stack size is only limited by the total SRAM size and the usage of the SRAM. All user programs must initialize the SP in the Reset routine (before subroutines or interrupts are executed). The Stack Pointer (SP) is read/write accessible in the I/O space. The data SRAM can easily be accessed through the five different addressing modes supported in the AVR architecture.

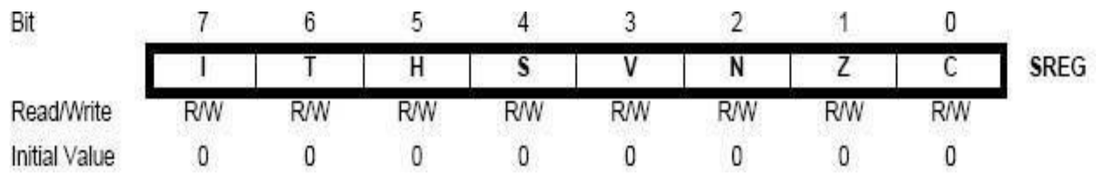
ALU – ARITHMETIC LOGIC UNIT

The high-performance AVR ALU operates in direct connection with all the 32 general purpose working registers. Within a single clock cycle, arithmetic operations between general purpose registers or between a register and an immediate are executed. The ALU operations are divided into three main categories – arithmetic, logical, and bit functions. Some implementations of the architecture also provide a powerful multiplier supporting both signed/unsigned multiplication and fractional format. See the "Instruction Set" section for a detailed description.

STATUS REGISTER

The Status Register contains information about the result of the most recently executed arithmetic instruction. This information can be used for altering program flow in order to perform conditional operations. Note that the Status Register is updated after all ALU operations, as specified in the Instruction Set Reference. This will in many cases remove the need for using the dedicated compare instructions, resulting in faster and more compact code. The Status Register is not automatically stored when entering an interrupt routine and restored when returning from an interrupt. This must be handled by software.

The AVR Status Register – SREG is defined as:



Bit	7	6	5	4	3	2	1	0	
	I	T	H	S	V	N	Z	C	SREG
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Fig 5.7: AVR status register

Bit 7 – I: Global Interrupt Enable

The Global Interrupt Enable bit must be set for the interrupts to be enabled. The individual interrupt enable control The Global Interrupt Enable bit must be set for the interrupts to be enabled. The individual interrupt enable control is then performed in separate control registers. If the Global Interrupt Enable Register is cleared, none of the interrupts are enabled independent of the individual interrupt enable settings. The I-bit is cleared by hardware after an interrupt has occurred, and is set by the RETI instruction to enable subsequent interrupts. The I-bit can also be set and cleared by the application with the SEI and CLI instructions, as described in the instruction set reference.

Bit 6 – T: Bit Copy Storage

The Bit Copy instructions BLD (Bit Load) and BST (Bit Store) use the T-bit as source or destination for the operated bit. A bit from a register in the Register File can be copied into T by the BST instruction, and a bit in T can be copied into a bit in a register in the Register File by the BLD instruction.

Bit 5 – H: Half Carry Flag

The Half Carry Flag H indicates a Half Carry in some arithmetic operations The Half Carry Flag H indicates a Half Carry in some arithmetic operations. Half Carry Is useful in BCD arithmetic. See the “Instruction Set Description” for detailed information.

Bit 4 – S: Sign Bit

The S-bit is always an exclusive or between the Negative Flag N and the Two’s Complement Overflow Flag V. See the “Instruction Set Description” for detailed information.

Bit 3 – V: Two’s Complement Overflow Flag

The Two’s Complement Overflow Flag V supports two’s complement arithmetic.

Bit 2 – N: Negative Flag

The Negative Flag N indicates a negative result in an arithmetic or logic operation.

Bit 1 – Z: Zero Flag

The Zero Flag Z indicates a zero result in an arithmetic or logic operation.

Bit 0 – C: Carry Flag

The Carry Flag C indicates a carry in an arithmetic or logic operation.

STACK POINTER

The Stack is mainly used for storing temporary data, for storing local variables and for storing return addresses after interrupts and subroutine calls. Note that the Stack is implemented as growing from higher to lower memory locations. The Stack Pointer Register always points to the top of the Stack. The Stack Pointer points to the data SRAM Stack area where the Subroutine and Interrupt Stacks are located. A Stack PUSH command will decrease the Stack Pointer.

The Stack in the data SRAM must be defined by the program before any subroutine calls are executed or interrupts are enabled. Initial Stack Pointer value equals the last address of the internal SRAM and the Stack Pointer must be set to point above start of the SRAM.

The AVR ATmega128A Stack Pointer is implemented as two 8-bit registers in the I/O space. The number of bits actually used is implementation dependent. Note that the data space in some implementations of the AVR architecture is so small that only SPL is needed. In this case, the SPH Register will not be present.SPH and SPL - Stack Pointer High and Low Register.

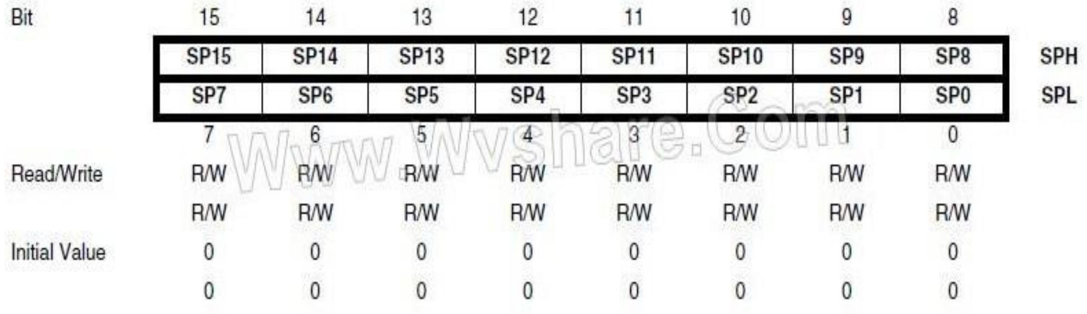
Table 5.2.1 Stack Pointer instructions

Instruction	Stack pointer	Description
PUSH	Decremented by 1	Data is pushed onto the stack
CALL , ICALL RCALL	Decremented by 2	Return address is pushed onto the stack with a subroutine call or interrupt
POP	Incremented by 1	Data is popped from the stack
RET RETI	Incremented by 2	Return address is popped from the stack with return from subroutine or return from interrupt

INTERRUPT RESPONSE TIME

The interrupt execution response for all the enabled AVR interrupts is four clock cycles minimum. After four clock cycles the program vector address for the actual interrupt handling routine is executed.

During this four clock cycle period, the Program Counter is pushed onto the Stack. The vector is normally a jump to the interrupt routine, and this jump takes three clock cycles. If an interrupt occurs during execution of a multi-cycle instruction, this instruction is completed before the interrupt is served. If an interrupt occurs when the MCU is in sleep mode, the interrupt execution response time is increased by four clock cycles. This increase comes in addition to the start-up time from the selected sleep mode.



Bit	15	14	13	12	11	10	9	8	
	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8	SPH
	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0	SPL
	7	6	5	4	3	2	1	0	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

Fig 5.8: SPH and SPL - Stack Pointer High and Low Register

AVR Memories

This section describes the different memories in the ATmega328. The AVR architecture has two main memory spaces, the Data Memory and the Program Memory space. In addition, theATmega328 features an EEPROM Memory for data storage. All three memory spaces are linear and regular.

In-System Reprogrammable Flash Program Memory:

The ATmega328 contains 4/8/16/32Kbytes On-chip In-System Reprogrammable Flash memory for program storage. Since all AVR instructions are 16 or 32 bits wide, the Flash is organized as 2/4/8/16K x 16. For software security, the Flash Program memory space is divided into two sections, Boot Loader Section and Application Program Section. The Flash memory has an endurance of at least 10,000 write/erase cycles. The ATmega328 Program Counter (PC) is 11/12/13/14 bits wide, thus addressing the 2/4/8/16K program memory locations.

SRAM Data Memory:

ATmega328 is a complex microcontroller with more peripheral units than can be supported within the 64 locations reserved in the Opcode for the IN and OUT instructions. For the Extended I/O space from 0x60 - 0xFF in SRAM, only the ST/STS/STD and LD/LDS/LDD instructions can be used.

The lower 768/1280/1280/2303 data memory locations address both the Register File, the I/O memory, Extended I/O memory, and the internal data SRAM. The first 32 locations address the Register File, the next 64 location the standard I/O memory, then 160 locations of Extended I/O memory, and the next 512/1024/1024/2048 locations address the internal data SRAM. The five different addressing modes for the data memory cover: Direct, Indirect with Displacement, Indirect, Indirect with Pre-decrement, and Indirect with Post-increment. In The Register File, Registers R26 to R31 Feature the indirect addressing pointer registers. The direct addressing reaches the entire data space. The Indirect with Displacement mode reaches 63 address locations from the base address given by the Y- or Z register.

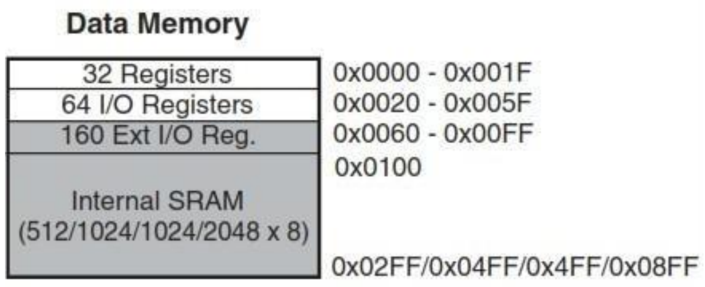


Fig 5.9: Data Memory Map

When using register indirect addressing modes with automatic pre-decrement and postincrement, the address registers X, Y, and Z are decremented or incremented. The 32 general purpose working registers, 64 I/O Registers, 160 Extended I/O Registers, and the 512/1024/1024/2048 bytes of internal data SRAM in the ATmega328 are all accessible through all these addressing modes.

INTERRUPTS

This section describes the specifics of the interrupt handling as performed in the Atmega328. In Atmega328Each Interrupt Vector occupies two instruction words and the Reset Vector is affected by the BOOTRST fuse, and the Interrupt Vector start address is affected by the IVSEL bit in MCUCR.

When the IVSEL bit in MCUCR is set, Interrupt Vectors will be moved to the start of the Boot Flash Section. The address of each Interrupt Vector will then be the address in this table added to the start address of the Boot Flash Section.Table below shows reset and Interrupt Vectors placement for the various combinations of BOOTRST and IVSEL settings. The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to versionR2) programmed as a USB-to-serial converter.

Table 5. 2.2 Reset and Interrupt Vectors in ATMEGA 328 and ATMEGA 328P

Vector No.	**Program Address**	**Source**	**Interrupt Definition**
1	0x0000	RESET	External Pin, Power-on Reset, Brown-out Reset and Watchdog System Reset
2	0x0002	INT0	External Interrupt Request 0
3	0x0004	INT1	External Interrupt Request 0
4	0x0006	PCINTO	Pin Change Interrupt Request 0
5	0x0008	PCINT1	Pin Change Interrupt Request 1
6	0x000A	PCINT2	Pin Change Interrupt Request 2
7	0x000C	WDT	Watchdog Time-out Interrupt
8	0x000E	TIMER2 COMPA	Timer/Counter2 Compare Match A
9	0x0010	TIMER2 COMPB	Timer/Counter2 Compare Match B
10	0x0012	TIMER2 OVF	Timer/Counter 2 Overflow

11	0x0014	TIMER1 CAPT	Timer/Counter 2 Capture Event
12	0x0016	TIMER1 COMPA	Timer/Counter1 Compare Match A
13	0x0018	TIMER1 COMPB	Timer/Counter1 Compare Match B
14	0x001A	TIMER 1 OVF	Timer/Counter1 Overflow
15	0x001C	TIMER0 COMPA	Timer/Counter0 Compare Match A
16	0x001E	TIMER0 COMPB	Timer/Counter0 Compare Match B
17	0x0020	TIME0 OVF	Timer/Counter0 Overflow
18	0x0022	SPI, STC	SPI Serial Transfer Complete
19	0x0024	USART, RX	USART RX Complete
20	0x0026	USART, UDRE	USART, Data Register Empty
21	0x0028	USART, TX	USART, TX Complete
22	0x002A	ADC	ADC Conversion Complete
23	0x002C	EE READY	EEPROM Ready
24	0x002E	ANALOG COMP	Analog Comparator
25	0x0030	TWI	2-wire Serial Interface
26	0x0032	SPM READY	Store Program Memory Ready

If the program never enables an interrupt source, the Interrupt Vectors are not used, and regular program code can be placed at these locations. This is also the case if the Reset Vector is in the Application section while the Interrupt Vectors are in the Boot section or vice versa.

Table 5.2.3 Reset and Interrupt Vectors Placement in ATmega328 and ATmega328P

BOOTRST	IVSEL	Reset Address	Interrupt Vectors Start Address
1	0	0x000	0x002
1	1	0x000	Boot Reset Address + 0x0002
0	0	BootReset Address	0x002
0	1	BootReset Address	Boot Reset Address + 0x002

Arduino with ATmega328

The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button.

It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

- Pin out: Added SDA and SCL pins that are near to the AREF pin and two other new pins placed near to the RESET pin, the IOREF that allow the shields to adapt to the voltage provided from the board. In future, shields will be compatible with both the board that uses the AVR, which operates with 5V and with the Arduino. Due that operates with 3.3V. The second one is a not connected pin that is reserved for future purposes.
- Stronger RESET circuit.
- Atmega 16U2 replace the 8U2.

"Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version 1.0 will be the reference versions of Arduino, moving forward. The Uno is the latest in a series of USB Arduino boards, and the reference model for the Arduino platform; for a comparison with previous versions, see the index of Arduino boards.

Arduino Characteristics Power:

The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector. The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The power pins are as follows:

- **VIN:** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V:** This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board. We don't advise it.

- **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND.** Ground pins.
- **IOREF.** This pin on the Arduino board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs for working with the 5V or 3.3V.

Memory:

The ATmega328 has 32 KB (with 0.5 KB used for the boot loader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

Serial Communication:

A Software serial library allows for serial communication on any of the Uno's digital pins. The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus. For SPI communication, use the SPI library.

5.2.2 Block Diagram

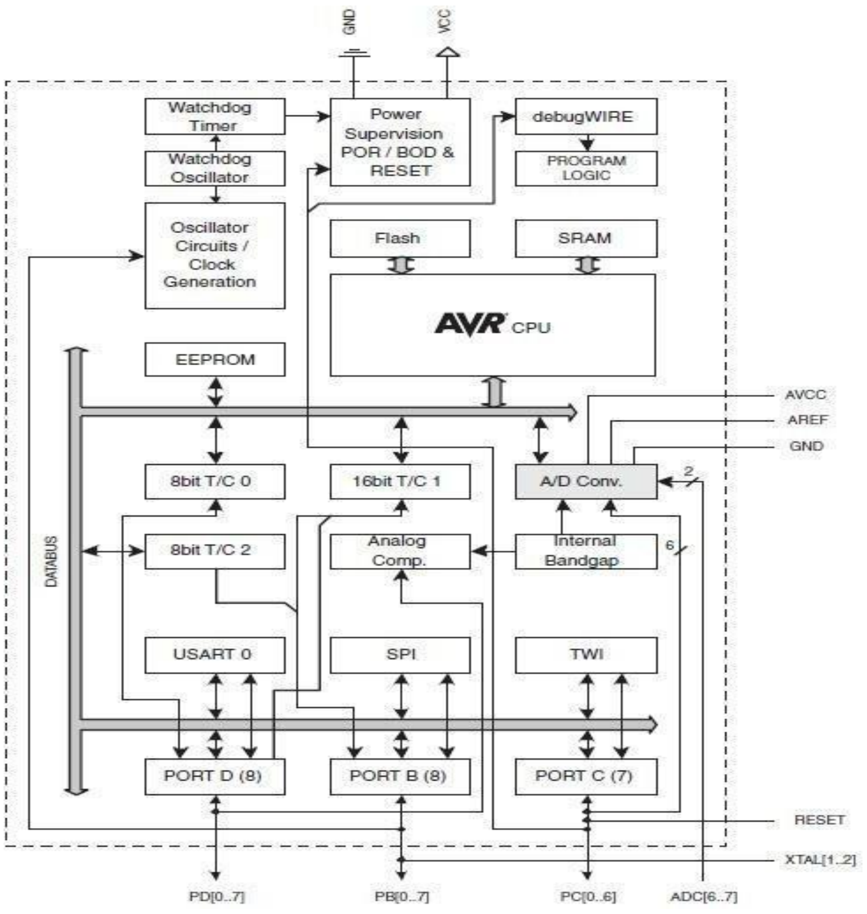


Fig 5.10: Arduino Block Diagram

5.2.3 Introduction To GPS Module

The GPS module is integrated into the system to provide real-time tracking of the vehicle’s location. The GPS module sends continuous data regarding the vehicle's position (latitude, longitude, and altitude) to the Arduino microcontroller.

1. **Location Data Acquisition**: The GPS module, often based on u-blox or Neo-6M technology, is configured to constantly collect location data at regular intervals (e.g., every 5 seconds or 1 minute). The Arduino processes this data and sends it through the GPS modem to the user for real-time vehicle tracking.

2. **Geofencing**: The system can use geofencing features, where the user can set a virtual boundary around a specific location (e.g., a city or parking area). If the vehicle crosses this boundary, the Arduino sends an alert via the GPS modem, notifying the vehicle owner of unauthorized movement.

5.2.4 Block Diagram

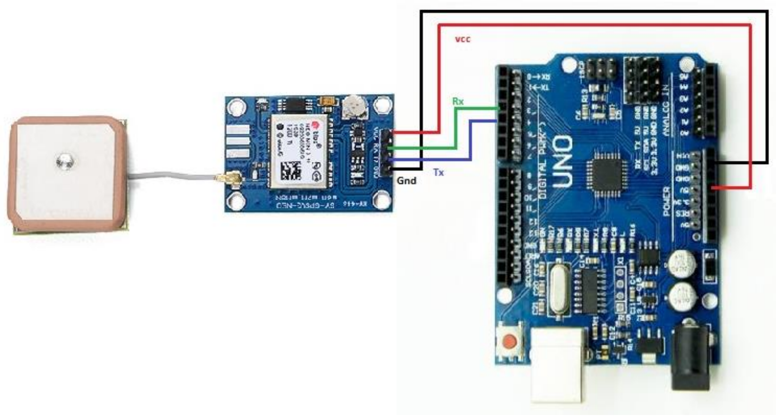


Fig 5.11: GPS Module

5.2.5 Introduction To GPRS Modem

The GPS modem, typically based on GSM or GPRS technology, is crucial for enabling communication between the vehicle and the vehicle owner or fleet manager. It allows for remote monitoring and control of the vehicle.

1. **Location Transmission**: The Arduino transmits the vehicle’s real-time GPS location data through the GPS modem to the owner’s mobile phone or web-based platform. This enables the owner to track the vehicle’s movements continuously via SMS or an app.

2. **Remote Control and Alerts:** In addition to location updates, the GPS modem is used to send real-time alerts in case of emergencies (e.g., theft, abnormal vehicle movement). The

owner can also send commands to remotely immobilize the vehicle, stop the engine, or activate safety features like the alarm system.

3. Communication Modules: If the system is designed for remote access or notifications, communication devices (such as Bluetooth or Wi-Fi modules) are incorporated. Logic devices manage the flow of information between the car security system and a mobile application or monitoring service.

5.2.6 Block Diagram

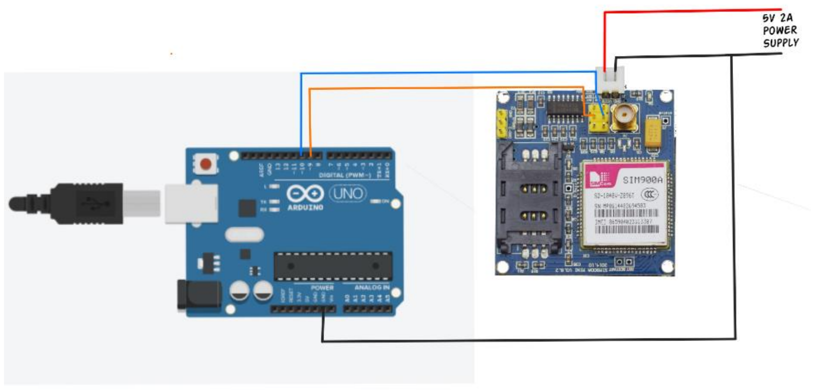


Fig 5.12: GSM module

5.2.7 Introduction To MEMS

MEMS, or Microelectromechanical Systems, refer to tiny devices that combine mechanical and electrical components at a micro-scale. These systems are typically fabricated using microfabrication techniques similar to those used in the semiconductor industry, allowing them to be integrated with electronic circuits. MEMS devices can perform a wide range of functions, including sensing, actuation, and signal processing.

Key Components of MEMS:

1. **Micro Sensors**: MEMS sensors detect physical properties like acceleration, pressure, temperature, or chemical composition. These sensors are widely used in applications such as automotive airbags, smartphones, and medical devices.
2. **Micro Actuators**: These are components that can perform mechanical tasks, such as moving a part or generating force. They can control tiny movements based on electrical signals.
3. **Micro Structures**: MEMS devices often include micro-scale structures, such as beams, membranes, or cantilevers, which interact with the environment to detect or create mechanical forces.

4. **Microelectronics**: MEMS systems integrate microelectronics to process the data received from sensors or to control actuators. This allows for high-performance systems that are compact and energy-efficient.

Key Advantages of MEMS:

- **Miniaturization**: MEMS devices are extremely small, often measuring only a few millimeters or micrometers in size.
- **Low Power Consumption**: Due to their tiny size and efficient design, MEMS systems often consume very little power, making them ideal for portable and battery-operated devices.
- **Integration with Electronics**: MEMS can be easily integrated with existing electronic systems to provide enhanced functionality.
- **High Sensitivity and Precision**: MEMS sensors and actuators can detect and control very small physical changes with high accuracy.

5.2.8 Block Diagram

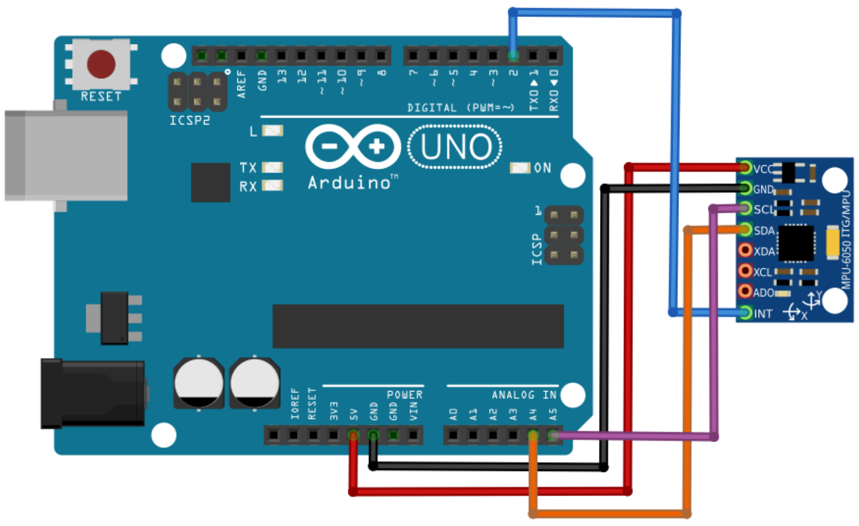


Fig 5.13: MEMS SENSOR

5.2.9 Additional Sensors For Enhanced Security

For a more comprehensive vehicle protection system, additional sensors can be integrated into the design, such as:

- **Heartbeat Sensor:** A heartbeat sensor, also known as a heart rate sensor, is a device used to measure the number of heartbeats per minute (BPM) to monitor an individual's heart activity. It works by detecting the changes in blood flow caused by the beating of the heart. These sensors are commonly found in health and fitness devices like wearables (e.g., fitness trackers and smartwatches), medical monitoring equipment, and even smartphones.

- **Temperature Sensors**: Used to monitor the engine temperature and prevent overheating. If the vehicle's engine temperature exceeds a certain threshold, the system will alert the vehicle owner and provide real-time updates.

CHAPTER – 6

RESULTS

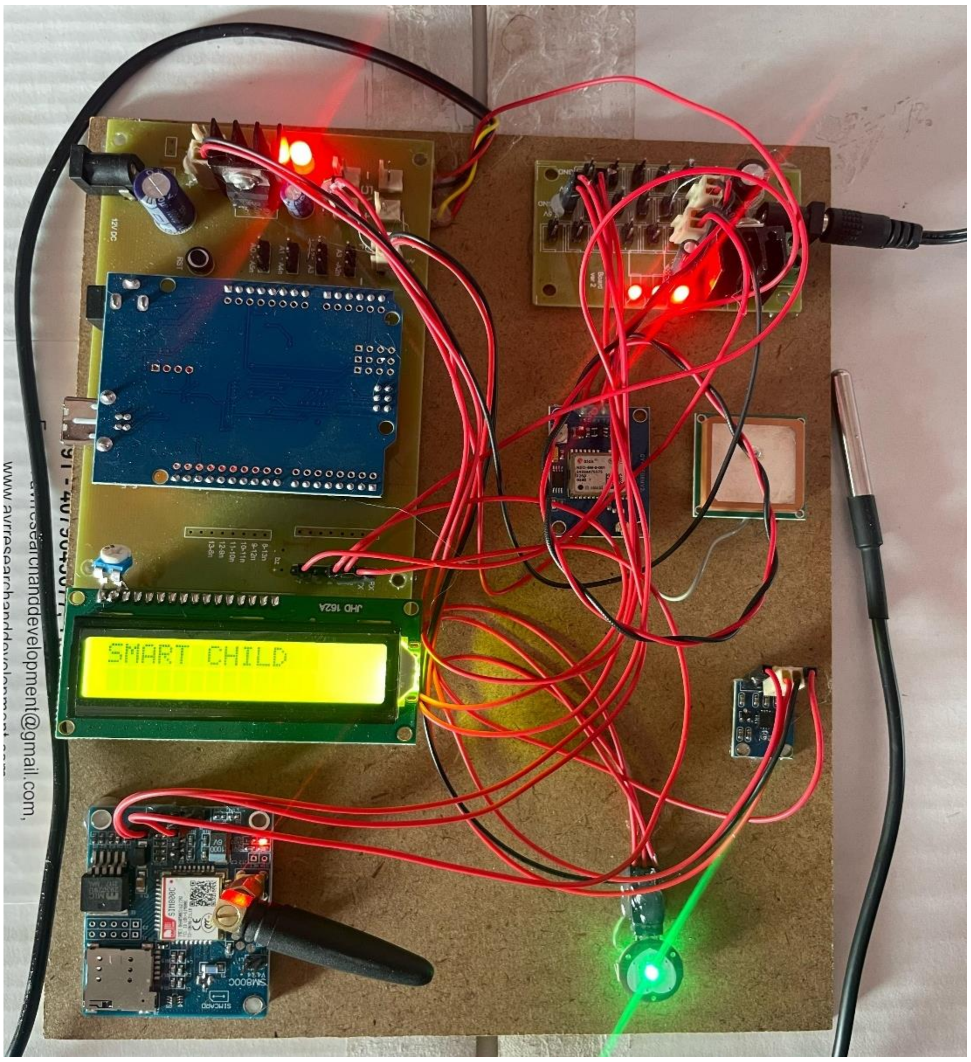
A wearable device for child safety that integrates GPS, GSM, and MEMS sensors offers a powerful and comprehensive solution to ensuring the well-being of children. The integration of these technologies works together to provide continuous monitoring, real-time communication, and a heightened level of safety for children in various environments.

The GPS module enables the wearable to track the child’s location in real-time, offering parents the ability to monitor their child's movements throughout the day. This is particularly useful in busy or crowded areas, such as shopping malls, parks, or school environments, where it might be easy for a child to wander off. Parents can set virtual boundaries, called geofences, around safe areas like the home, school, or park. If the child moves beyond these boundaries, the device immediately sends an alert to the parent’s smartphone, ensuring they are notified if their child is in an unsafe or unexpected location. This feature provides a critical layer of security by helping parents stay informed and react quickly.

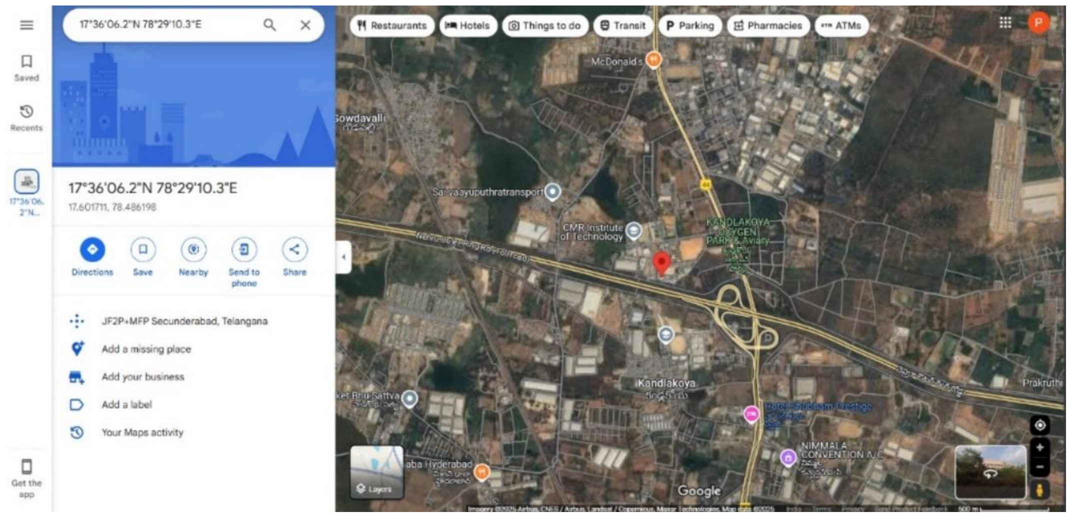
The GSM module adds an essential communication feature to the device, allowing the child to stay in touch with their parents at any time. Through GSM connectivity, the device can facilitate voice calls and SMS alerts, enabling two-way communication between the parent and child. In the event of an emergency or if the child feels unsafe, they can use the SOS button on the device, which will immediately send an emergency message or call to the parent’s phone. This ability to quickly contact the parent or emergency services adds an extra layer of reassurance, especially in situations where the child may not be able to communicate verbally. Additionally, GSM enables the transmission of location data, ensuring that parents can access the child’s real-time location anytime via an app, further enhancing the device’s functionality.

The inclusion of MEMS sensors (Microelectromechanical Systems) adds advanced sensing capabilities to the wearable. These sensors typically include accelerometers and gyroscopes, which can detect physical changes in the child’s environment. For example, accelerometers can sense sudden movements or impact, such as a fall or a jolt, and instantly trigger an alert to the parent’s smartphone. This is particularly valuable in cases where a child might fall or experience an accident, allowing the parent to respond quickly and provide assistance. Moreover, these sensors can monitor the child’s activity levels, detecting if the child is engaged in strenuous activity or if something unusual happens, such as an abrupt stop or an unexpected movement.

MEMS sensors can also be used to detect environmental hazards, such as high noise levels or physical disturbances, providing additional protection by alerting the parent to potentially unsafe situations.


SMART CHILD
www.avrresearchanddevelopment.com
avrresearchanddevelopment@gmail.com,

The combination of GPS, GSM, and MEMS sensors in a wearable device offers an all-encompassing safety solution. The GPS ensures accurate location tracking, the GSM enables real-time communication and emergency alerts, and the MEMS sensors provide monitoring of physical activity and environmental factors. Together, these technologies work seamlessly to provide parents with constant awareness of their child's location, health, and safety, offering peace of mind and confidence in knowing they can act quickly in the event of any emergencies. This integration makes the wearable device an invaluable tool for child safety, ensuring children are not only protected from external dangers but also connected to their parents in real time.



ADVANTAGES

1. Track Where Your Child Is (GPS + GSM)

- Parents can see where their child is at all times using GPS.
- If the child leaves a safe area, the device sends an alert to the parents.

2. Emergency Alerts (GSM)

- The child can press an SOS button to send an emergency alert to their parents, showing their location.

3. Monitor Your Child's Health (Heartbeat & Temperature Sensors)

- The device checks the child's heart rate and body temperature.
- If something's wrong, like a fever or high heart rate, the parents are notified.

4. Detect Accidents (MEMS)

- If the child falls or has an accident, the device can send an alert to the parents.
- It can also track the child's activity to make sure they're okay.

APPLICATIONS

Wearable devices for child safety have many practical applications that help keep kids safe and provide peace of mind for parents. These devices allow parents to track their child's location in real-time, whether they're at school, the park, or playing outside, helping prevent them from getting lost in crowded places. In case of an emergency, the child can press an SOS button to immediately send an alert to the parents with their exact location. The device also monitors the child's health, checking heart rate and body temperature, and alerts parents if something's wrong, like a fever or abnormal heart rate. If the child falls or gets hurt, the device can detect the accident and send an alert to the parents. Parents can set safe zones (like home or school) using geo-fencing, and if the child moves outside these areas, the device sends an alert. It also tracks the child’s activity, such as at school or during sports, ensuring they are safe while active. Additionally, the device can monitor the child’s body temperature, alerting parents if it becomes dangerously high or low. For young children, the wearable allows two-way communication, so parents can contact them easily. These devices are especially helpful for parents traveling, as they can continue to monitor their child’s location and health. For children with special needs, the device can send alerts if there are any health concerns or risky behaviors. Overall, these wearable devices help parents ensure their child’s safety and well-being while allowing the child to feel more independent and secure.

CONCLUSION

In conclusion, wearable devices for child safety offer a comprehensive solution for monitoring and protecting children in various situations. By combining GPS tracking, health monitoring, emergency alerts, and features like fall detection and geo-fencing, these devices give parents peace of mind while ensuring their child's well-being. Whether it's keeping track of their location, responding to an emergency, or monitoring health conditions, these devices help parents stay informed and ready to act. They also provide children with a sense of independence, knowing that their safety is always being looked after. As technology continues to evolve, these wearables will only become more advanced, making it easier for parents to protect their children while fostering a sense of freedom and security for kids.

FUTURE SCOPE

The future of wearable devices for child safety holds tremendous promise. With advancements in health monitoring, AI, communication features, and integration with smart home systems, these devices will become even more powerful tools for keeping children safe. They will not only track location and monitor health but also provide proactive alerts, improve communication, and ensure more comfortable and secure experiences for both parents and children. As technology continues to evolve, the possibilities for these devices are nearly limitless, and they will become an even more essential part of child safety and well-being.

REFERENCES

[1] SSB. Mallick and A. K. Patro, \"Heart Rate Monitoring System Using Finger Tip Through Arduino And Processing Software\", International Journal of Science Engineering and Technology Research (IJSETR), vol. 5, no. 1, January 2016, ISSN 2278-7798.

[2] Dustin T. Weiler, Stefanie O. Villajuan, Laura Edkins, Sean Cleary and Jason J. Saleem, \"Wearable Heart Rate Monitor Technology Accuracy in Research: A Comparative Study between PPG and ECG Technology\", Proceedings of the Human Factors and Ergonomics Society 2017 Annual Meeting.

[3] Marius Valerian Paulet, Oana Maria Neacsu and Andrei Salceanu, \"Wireless monitoring system of the heart rate\", 2014 International Conference and Exposition on Electrical and Power Engineering (EPE), ISBN 978-1-4799- 5849-8.

[4] Kala Venugopal and Amit Kumar, \"Centralized Heart Rate Monitoring and Automated Message Alert System using WBAN\", International Journal of Scientific and Research Publications, vol. 3, no. 9, September 2013, ISSN 2250-3153.

[5] Kainat Zeba, Lakshmi S Patil, Sanjana R Gowda, R Varsha and Shobha Chandra K, \"Real Time Heart Attack and Heart Rate Monitoring Android Application\", International Journal of Computer Science and Mobile Computing, vol. 7, no. 4, pp. 115-124, April 2018, ISSN 2320-088X.

[6] Yuan-Hsiang Lin, I-Chien Jan, P. C. -. Ko, Yen-Yu Chen, Jau-Min Wong and Gwo-Jen Jan, \"A wireless PDA-based physiological monitoring system for patient transport\", IEEE Transactions on Information Technology in Biomedicine, vol. 8, no. 4, pp. 439-447, Dec. 2004.

[7] H. Ren, H. Jin, C. Chen, H. Ghayvat and W. Chen, \"A Novel Cardiac Auscultation Monitoring System Based on Wireless Sensing for Healthcare\", IEEE Journal of Translational Engineering in Health and Medicine, vol. 6, pp. 1-12, 2018.

[8] A. Hodge, H. Humnabadkar and A. Bidwai, \"Wireless Heart Rate Monitoring and Vigilant System\", 2018 3rd International Conference for Convergence in Technology (I2CT), pp. 1-5, 2018.